SCIENTIFIC REPORTS
nature research

OPEN

Training Optimization for Gate-Model Quantum Neural Networks

Laszlo Gyongyosi[1,2,3] & Sandor Imre[2]

Received: 24 July 2018
Accepted: 15 August 2019
Published online: 03 September 2019

Gate-based quantum computations represent an essential to realize near-term quantum computer architectures. A gate-model quantum neural network (QNN) is a QNN implemented on a gate-model quantum computer, realized via a set of unitaries with associated gate parameters. Here, we define a training optimization procedure for gate-model QNNs. By deriving the environmental attributes of the gate-model quantum network, we prove the constraint-based learning models. We show that the optimal learning procedures are different if side information is available in different directions, and if side information is accessible about the previous running sequences of the gate-model QNN. The results are particularly convenient for gate-model quantum computer implementations.

Gate-based quantum computers represent an implementable way to realize experimental quantum computations on near-term quantum computer architectures[1–13]. In a gate-model quantum computer, the transformations are realized by quantum gates, such that each quantum gate is represented by a unitary operation[14–26]. An input quantum state is evolved through a sequence of unitary gates and the output state is then assessed by a measurement operator[14–17]. Focusing on gate-model quantum computer architectures is motivated by the successful demonstration of the practical implementations of gate-model quantum computers[7–11], and several important developments for near-term gate-model quantum computations are currently in progress. Another important aspect is the application of gate-model quantum computations in the near-term quantum devices of the quantum Internet[27–43].

A quantum neural network (QNN) is formulated by a set of quantum operations and connections between the operations with a particular weight parameter[14,25,26,44–47]. Gate-model QNNs refer to QNNs implemented on gate-model quantum computers[14]. As a corollary, gate-model QNNs have a crucial experimental importance since these network structures are realizable on near-term quantum computer architectures. The core of a gate-model QNN is a sequence of unitary operations. A gate-model QNN consists of a set of unitary operations and communication links that are used for the propagation of quantum and classical side information in the network for the related calculations of the learning procedure. The unitary transformations represent quantum gates parameterized by a variable referred to as gate parameter (weight). The inputs of the gate-model QNN structure are a computational basis state and an auxiliary quantum system that serves a readout state in the output measurement phase. Each input state is associated with a particular label. In the modeled learning problem, the training of the gate-model QNN aims to learn the values of the gate parameters associated with the unitaries so that the predicted label is close to a true label value of the input (i.e., the difference between the predicted and true values is minimal). This problem, therefore, formulates an objective function that is subject to minimization. In this setting, the training of the gate-model QNN aims to learn the label of a general quantum state.

In artificial intelligence, machine learning[4–6,19,23,45,46,48–53] utilizes statistical methods with measured data to achieve a desired value of an objective function associated with a particular problem. A learning machine is an abstract computational model for the learning procedures. A constraint machine is a learning machine that works with constraint, such that the constraints are characterized and defined by the actual environment[48].

The proposed model of a gate-model quantum neural network assumes that quantum information can only be propagated forward direction from the input to the output, and classical side information is available via classical links. The classical side information is processed further via a post-processing unit after the measurement of the output. In the general gate-model QNN scenario, it is assumed that classical side information can be propagated arbitrarily in the network structure, and there is no available side information about the previous running sequences of the gate-model QNN structure. The situation changes, if side information propagates only backward

[1]School of Electronics and Computer Science, University of Southampton, Southampton, SO17 1BJ, UK. [2]Department of Networked Systems and Services, Budapest University of Technology and Economics, Budapest, H-1117, Hungary. [3]MTA-BME Information Systems Research Group, Hungarian Academy of Sciences, Budapest, H-1051, Hungary. Correspondence and requests for materials should be addressed to L.GY. (email: l.gyongyosi@soton.ac.uk)

direction and side information about the previous running sequences of the network is also available. The resulting network model is called gate-model recurrent quantum neural network (RQNN).

Here, we define a constraint-based training optimization method for gate-model QNNs and RQNNs, and propose the computational models from the attributes of the gate-model quantum network environment. We show that these structural distinctions lead to significantly different computational models and learning optimization. By using the constraint-based computational models of the QNNs, we prove the optimal learning methods for each network—nonrecurrent and recurrent gate-model QNNs—vary. Finally, we characterize optimal learning procedures for each variant of gate-model QNNs.

The novel contributions of our manuscript are as follows.

- *We study the computational models of nonrecurrent and recurrent gate-model QNNs realized via an arbitrary number of unitaries.*
- *We define learning methods for nonrecurrent and recurrent gate-model QNNs.*
- *We prove the optimal learning for nonrecurrent and recurrent gate-model QNNs.*

This paper is organized as follows. In Section 2, the related works are summarized. Section 3 defines the system model and the parameterization of the learning optimization problem. Section 4 proves the computational models of gate-model QNNs. Section 5 provides learning optimization results. Finally, Section 6 concludes the paper. Supplemental information is included in the Appendix.

Related Works

Gate-model quantum computers. A theoretical background on the realizations of quantum computations in a gate-model quantum computer environment can be found in[15] and[16]. For a summary on the related references[1–3,13,15–17,54,55], we suggest[56].

Quantum neural networks. In[14], the formalism of a gate-model quantum neural network is defined. The gate-model quantum neural network is a quantum neural network implemented on gate-model quantum computer. A particular problem analyzed by the authors is the classification of classical data sets which consist of bitstrings with binary labels.

In[44], the authors studied the subject of quantum deep learning. As the authors found, the application of quantum computing can reduce the time required to train a deep restricted Boltzmann machine. The work also concluded that quantum computing provides a strong framework for deep learning, and the application of quantum computing can lead to significant performance improvements in comparison to classical computing.

In[45], the authors defined a quantum generalization of feedforward neural networks. In the proposed system model, the classical neurons are generalized to being quantum reversible. As the authors showed, the defined quantum network can be trained efficiently using gradient descent to perform quantum generalizations of classical tasks.

In[46], the authors defined a model of a quantum neuron to perform machine learning tasks on quantum computers. The authors proposed a small quantum circuit to simulate neurons with threshold activation. As the authors found, the proposed quantum circuit realizes a "œquantum neuron". The authors showed an application of the defined quantum neuron model in feedforward networks. The work concluded that the quantum neuron model can learn a function if trained with superposition of inputs and the corresponding output. The proposed training method also suffices to learn the function on all individual inputs separately.

In[25], the authors studied the structure of artificial quantum neural network. The work focused on the model of quantum neurons and studied the logical elements and tests of convolutional networks. The authors defined a model of an artificial neural network that uses quantum-mechanical particles as a neuron, and set a Monte-Carlo integration method to simulate the proposed quantum-mechanical system. The work also studied the implementation of logical elements based on introduced quantum particles, and the implementation of a simple convolutional network.

In[26], the authors defined the model of a universal quantum perceptron as efficient unitary approximators. The authors studied the implementation of a quantum perceptron with a sigmoid activation function as a reversible many-body unitary operation. In the proposed system model, the response of the quantum perceptron is parameterized by the potential exerted by other neurons. The authors showed that the proposed quantum neural network model is a universal approximator of continuous functions, with at least the same power as classical neural networks.

Quantum machine learning. In[57], the authors analyzed a Markov process connected to a classical probabilistic algorithm[58]. A performance evaluation also has been included in the work to compare the performance of the quantum and classical algorithm.

In[19], the authors studied quantum algorithms for supervised and unsupervised machine learning. This particular work focuses on the problem of cluster assignment and cluster finding via quantum algorithms. As a main conclusion of the work, via the utilization of quantum computers and quantum machine learning, an exponential speed-up can be reached over classical algorithms.

In[20], the authors defined a method for the analysis of an unknown quantum state. The authors showed that it is possible to perform "œquantum principal component analysis" by creating quantum coherence among different copies, and the relevant attributes can be revealed exponentially faster than it is possible by any existing algorithm.

In[21], the authors studied the application of a quantum support vector machine in Big Data classification. The authors showed that a quantum version of the support vector machine (optimized binary classifier) can be

implemented on a quantum computer. As the work concluded, the complexity of the quantum algorithm is only logarithmic in the size of the vectors and the number of training examples that provides a significant advantage over classical support machines.

In[22], the problem of quantum-based analysis of big data sets is studied by the authors. As the authors concluded, the proposed quantum algorithms provide an exponential speedup over classical algorithms for topological data analysis.

The problem of quantum generative adversarial learning is studied in[51]. In generative adversarial networks a generator entity creates statistics for data that mimics those of a valid data set, and a discriminator unit distinguishes between the valid and non-valid data. As a main conclusion of the work, a quantum computer allows us to realize quantum adversarial networks with an exponential advantage over classical adversarial networks.

In[54], super-polynomial and exponential improvements for quantum-enhanced reinforcement learning are studied.

In[55], the authors proposed strategies for quantum computing molecular energies using the unitary coupled cluster ansatz.

The authors of[56] provided demonstrations of quantum advantage in machine learning problems.

In[57], the authors study the subject of quantum speedup in machine learning. As a particular problem, the work focuses on finding Boolean functions for classification tasks.

System Model

Gate-model quantum neural network.

Definition 1 *A* QNN_{QG} *is a quantum neural network* (QNN) *implemented on a gate-model quantum computer with a quantum gate structure QG. It contains quantum links between the unitaries and classical links for the propagation of classical side information. In a* QNN_{QG}*, all quantum information propagates forward from the input to the output, while classical side information can propagate arbitrarily (forward and backward) in the network. In a* QNN_{QG}*, there is no available side information about the previous running sequences of the structure.*

Using the framework of[14], a QNN_{QG} is formulated by a collection of L unitary gates, such that an i-th, $i=1, \ldots, L$ unitary gate $U_i(\theta_i)$ is

$$U_i(\theta_i) = \exp(-i\theta_i P), \tag{1}$$

where P is a generalized Pauli operator formulated by a tensor product of Pauli operators $\{X, Y, Z\}$, while θ_i is referred to as the gate parameter associated with $U_i(\theta_i)$.

In QNN_{QG}, a given unitary gate $U_i(\theta_i)$ sequentially acts on the output of the previous unitary gate $U_{i-1}(\theta_{i-1})$, without any nonlinearities[14]. The classical side information of QNN_{QG} is used in calculations related to error derivation and gradient computations, such that side information can propagate arbitrarily in the network structure.

The sequential application of the L unitaries formulates a unitary operator $U(\overrightarrow{\theta})$ as

$$U(\overrightarrow{\theta}) = U_L(\theta_L)U_{L-1}(\theta_{L-1})\ldots U_1(\theta_1), \tag{2}$$

where $U_i(\theta_i)$ identifies an i-th unitary gate, and $\overrightarrow{\theta}$ is the gate parameter vector

$$\overrightarrow{\theta} = (\theta_1, \ldots, \theta_{L-1}, \theta_L)^T. \tag{3}$$

At (2), the evolution of the system of QNN_{QG} for a particular input system $|\psi, \phi\rangle$ is

$$|Y\rangle = U(\overrightarrow{\theta})|\psi\rangle|\phi\rangle = U(\overrightarrow{\theta})|z\rangle|1\rangle = U(\overrightarrow{\theta})|z, 1\rangle, \tag{4}$$

where $|Y\rangle$ is the $(n+1)$-length output quantum system, and $|\psi\rangle = |z\rangle$ is a computational basis state, where z is an n-length string

$$z = z_1 z_2 \ldots z_n, \tag{5}$$

where each z_i represents a classical bit with values

$$z_i \in \{-1, 1\}, \tag{6}$$

while the $(n+1)$-th quantum state is initialized as

$$|\varphi\rangle = |1\rangle, \tag{7}$$

and is referred to as the readout quantum state.

Objective function.

The $f(\overrightarrow{\theta})$ objective function subject to minimization is defined for a QNN_{QG} as

$$f(\overrightarrow{\theta}) = \langle\overrightarrow{\theta}|\mathcal{L}(x_0, \tilde{l}(z))|\overrightarrow{\theta}\rangle, \tag{8}$$

where $\mathcal{L}(x_0, \tilde{l}(z))$ is the loss function[14], defined as

$$\mathcal{L}(x_0, \tilde{l}(z)) = 1 - l(z)\tilde{l}(z), \tag{9}$$

where $\tilde{l}(z)$ is the predicted value of the binary label

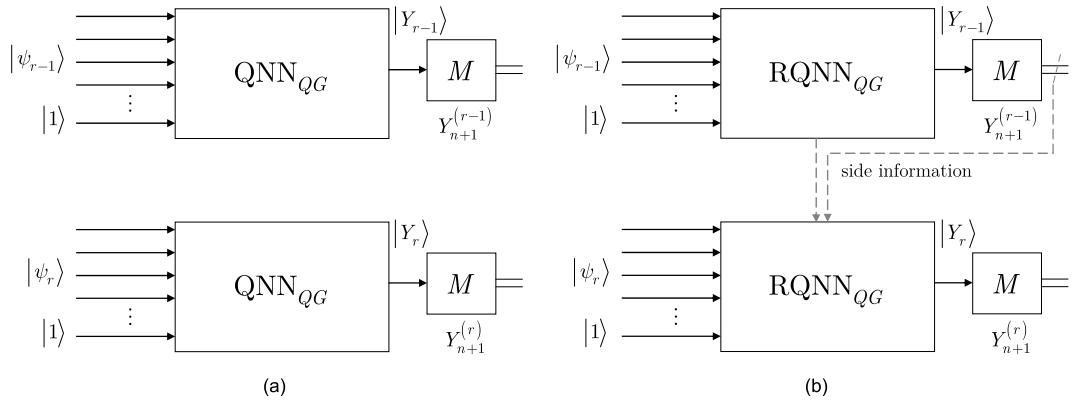

Figure 1. Schematic representation of a QNN$_{QG}$ and RQNN$_{QG}$ in an $(n-1)$-th and r-th measurement rounds. The $(n+1)$-length input systems of the $(r-1)$ and r-th measurement rounds are depicted by $|\psi_{r-1}\rangle|1\rangle$ and $|\psi_r\rangle|1\rangle$, the output systems are $|Y_{r-1}\rangle$ and $|Y_r\rangle$. The result of the M measurement operator in the $(n-1)$-th and r-th measurement rounds are denoted by $Y_{n+1}^{(r-1)}$ and $Y_{n+1}^{(r)}$. (**a**) In a QNN$_{QG}$, for an r-th measurement round, side information is not available about the previous $(n-1)$-th measurement round. (**b**) In a RQNN$_{QG}$, side information is available about the previous $(n-1)$-th measurement round in an r-th round.

$$l(z) \in \{-1, 1\} \tag{10}$$

of the string z, defined as[14]

$$\tilde{l}(z) = \langle z,1|(U(\overrightarrow{\theta}))^{\dagger} Y_{n+1} U(\overrightarrow{\theta})|z,\ 1\rangle, \tag{11}$$

where $Y_{n+1} \in \{-1, 1\}$ is a measured Pauli operator on the readout quantum state (7), while x_0 is as

$$x_0 = |z,\ 1\rangle. \tag{12}$$

The $\tilde{l}$ predicted value in (11) is a real number between -1 and 1, while the label $l(z)$ and Y_{n+1} are real numbers -1 or 1. Precisely, the $\tilde{l}$ predicted value as given in (11) represents an average of several measurement outcomes if Y_{n+1} is measured via R output system instances $|Y\rangle^{(r)}$-s, $r = 1, \ldots, R$[14].

The learning problem for a QNN$_{QG}$ is, therefore, as follows. At an $\mathcal{S}_T$ training set formulated via R input strings and labels

$$\mathcal{S}_T = \{z^{(r)},\ l(z^{(r)}),\ r = 1,\ \ldots,\ R\}, \tag{13}$$

where r refers to the r-th measurement round and R is the total number of measurement rounds, the goal is therefore to find the gate parameters (3) of the L unitaries of QNN$_{QG}$, such that $f(\overrightarrow{\theta})$ in (8) is minimal.

Recurrent Gate-model quantum neural network. **Definition 2** *An* RQNN$_{QG}$ *is a* QNN *implemented on a gate-model quantum computer with a quantum gate structure QG, such that the connections of* RQNN$_{QG}$ *form a directed graph along a sequence. It contains quantum links between the unitaries and classical links for the propagation of classical side information. In an* RQNN$_{QG}$, *all quantum information propagates forward, while classical side information can propagate only backward direction. In an* RQNN$_{QG}$, *side information is available about the previous running sequences of the structure.*

The classical side information of RQNN$_{QG}$ is used in error derivation and gradient computations, such that side information can propagate only in backward directions. Similar to the QNN$_{QG}$ case, in an RQNN$_{QG}$, a given i-th unitary $U_i(\theta_i)$ acts on the output of the previous unitary $U_{i-1}(\theta_{i-1})$. Thus, the quantum evolution of the RQNN$_{QG}$ contains no nonlinearities[14]. As follows, for an RQNN$_{QG}$ network, the objective function can be similarly defined as given in (8). On the other hand, the structural differences between QNN$_{QG}$ and RQNN$_{QG}$ allows the characterization of different computational models for the description of the learning problem. The structural differences also lead to various optimal learning methods for the QNN$_{QG}$ and RQNN$_{QG}$ structures as it will be revealed in Section 4 and Section 5.

Comparative representation. For a simple graphical representation, the schematic models of a QNN$_{QG}$ and RQNN$_{QG}$ for an $(r-1)$-th and r-th measurement rounds are compared in Fig. 1. The $(n+1)$-length input systems are depicted by $|\psi_{r-1}\rangle|1\rangle$ and $|\psi_r\rangle|1\rangle$, while the output systems are denoted by $|Y_{r-1}\rangle$ and $|Y_r\rangle$. The result of the M measurement operator in the $(r-1)$-th and r-th measurement rounds are denoted by $Y_{n+1}^{(r-1)}$ and $Y_{n+1}^{(r)}$. In Fig. 1(a), structure of a QNN$_{QG}$ is depicted for an $(r-1)$-th and r-th measurement round. In Fig. 1(b), the structure of a RQNN$_{QG}$ is illustrated. In a QNN$_{QG}$, side information is not available about the previous, $(r-1)$-th measurement round in a particular r-th measurement round. For an RQNN$_{QG}$, side information is available about the $(r-1)$-th measurement round (depicted by the dashed gray arrows) in a particular r-th measurement round.

The side information in the RQNN_{QG} setting refer to information about the gate-parameters and the measurement results of the $(r-1)$-th measurement round.

Parameterization. *Constraint machines.* The tasks of machine learning can be modeled via its mathematical framework and the constraints of the environment[4–6]. A $\mathcal{C}$ constraint machine is a learning machine working with constraints[48]. A constraint machine can be formulated by a particular function f or via some elements of a functional space $\mathcal{F}$. The constraints model the attributes of the environment of $\mathcal{C}$.

The learning problem of a $\mathcal{C}$ constraint machine can be represented via a $\mathcal{G} = (V, S)$ environmental graph[48,59-62]. The $\mathcal{G}$ environmental graph is a directed acyclic graph (DAG), with a set V of vertexes and a set S of arcs. The vertexes of $\mathcal{G}$ model associated features, while the arcs between the vertexes describe the relations of the vertexes.

The $\mathcal{G}$ environmental graph formalizes factual knowledge via modeling the relations among the elements of the environment[48]. In the environmental graph representation, the $\mathcal{C}$ constraint machine has to decide based on the information associated with the vertexes of the graph.

For any vertex v of V, a perceptual space element x, and its identifier $\langle x \rangle$ that addresses x in the computational model can be defined as a pair

$$(\langle x \rangle, x), \tag{14}$$

where $x \in \mathcal{X}$ is an element (vector) of the perceptual space $\mathcal{X} \subset \mathbb{C}^d$. Assuming that features are missing, the $\Diamond$ symbol can be used. Therefore, $\mathcal{X}$ is initialized as $\mathcal{X}_0$,

$$\mathcal{X}_0 = \mathcal{X} \bigcup \{\Diamond\}. \tag{15}$$

The environment is populated by individuals, and the $\mathcal{I}$ individual space is defined via V and $\mathcal{X}_0$ as

$$\mathcal{I} = V \times \mathcal{X}_0, \tag{16}$$

such that the existing features are associated with a subset $\widetilde{V}$ of V.

The features can be associated with the $\langle x \rangle$ identifier via a $f_{\mathcal{P}}$ perceptual map as

$$f_{\mathcal{P}} \colon \widetilde{V} \to \mathcal{X} \colon \ : x = f_{\mathcal{P}}(v). \tag{17}$$

If the condition

$$\forall\, v \in (V \backslash \widetilde{V}) \colon x = f_{\mathcal{P}}(v) = \Diamond \tag{18}$$

holds, then $f_{\mathcal{P}}$ is yielded as

$$f_{\mathcal{P}} \colon V \to \mathcal{X} \colon \ : x = f_{\mathcal{P}}(v). \tag{19}$$

A given individual $\iota \in \mathcal{I}$ is defined as a feature vector $x \in \mathcal{X}$. An $\iota \in \mathcal{I}$ individual of the individual space $\mathcal{I}$ is defined as

$$\iota = \Upsilon x + \neg\, \Upsilon v, \tag{20}$$

where $+$ is the sum operator in $\mathbb{C}^d$, $\neg$ is the negation operator, while Y is a constraint as

$$\Upsilon \colon (v \in \widetilde{V}) \vee (x \in \mathcal{X} \backslash \widetilde{\mathcal{X}}_0). \tag{21}$$

where $\mathcal{X}_0$ is given in (15). Thus, from (20), an individual ι is a feature vector x of $\mathcal{X}$ or a vertex v of $\mathcal{G}$.

Let $\iota^* \in \mathcal{I}$ be a specific individual, and let f be an agent represented by the function $f \colon \mathcal{I} \to \mathbb{C}^n$. Then, at a given environmental graph $\mathcal{G}$, the $\mathcal{C}$ constraint machine is defined via function f as a machine in which the learning and inference are represented via enforcing procedures on constraints C_{ι^*} and C_{ι}, such that for a $\mathcal{C}$ constraint machine the learning procedure requires the satisfaction of the constraints over all $\mathcal{I}^*$, while in the inference the satisfaction of the constraint is enforced over the given $\iota^* \in \mathcal{I}$[48], by theory. Thus, $\mathcal{C}$ is defined in a formalized manner, as

$$\mathcal{C} \equiv \begin{cases} C_{\iota} : \forall\, \iota \in \widetilde{\mathcal{I}} : \chi(v, f(\iota)) = 0, \\ C_{\iota_*} : \iota^* \in \mathcal{I} \backslash \widetilde{\mathcal{I}} : \chi(v^*, f^*(\iota^*)) = 0, \end{cases} \tag{22}$$

where $\widetilde{\mathcal{I}}$ is a subset of $\mathcal{I}$, ι^* refers to a specific individual, vertex or function, $\chi(\cdot)$ is a compact constraint function, while v^* and $f^*(\iota^*)$ refer to the vertex and function at ι^*, respectively.

Calculus of variations. Some elements from the calculus of variations[63,64] are utilized in the learning optimization procedure.

Euler-Lagrange Equations: The Euler-Lagrange equations are second-order partial differential equations with solution functions. These equations are useful in optimization problems since they have a differentiable functional that is stationary at the local maxima and minima[63]. As a corollary, they can be also used in the problems of machine learning.

Hessian Matrix: A Hessian matrix **H** is a square matrix of second-order partial derivatives of a scalar-valued function, or scalar field[63]. In theory, it describes the local curvature of a function of many variables. In a machine-learning setting, it is a useful tool to derive some attributes and critical points of loss functions.

Constraint-based Computational Model

In this section, we derive the computational models of the QNN_{QG} and RQNN_{QG} structures.

Environmental graph of a gate-model quantum neural network. **Proposition 1** *The* $\mathcal{G}_{\mathrm{QNN}_{QG}} = (V, S)$ *environmental graph of a* QNN_{QG} *is a DAG, where V is a set of vertexes, in our setting defined as*

$$V = \mathcal{S}_{in} \bigcup \mathcal{U} \bigcup \mathcal{Y}, \tag{23}$$

where $\mathcal{S}_{in}$ *is the input space,* $\mathcal{U}$ *is the space of unitaries,* $\mathcal{Y}$ *is the output space, and S is a set of arcs.*

Let $\mathcal{G}_{\mathrm{QNN}_{QG}}$ be an environmental graph of QNN_{QG}, and let v_{U_i} be a vertex, such that $v_{U_i} \in V$ is related to the unitary $U_i(\theta_i)$, where index $i = 0$ is associated with the $|z, 1\rangle$ input system with vertex v_0. Then, let v_{U_i} and v_{U_j} be connected vertices via directed arc s_{ij}, $s_{ij} \in S$, such that a particular θ_{ij} gate parameter is associated with the forward directed arc (Note: the notation $U_j(\theta_{ij})$ refers to the selection of θ_j for the unitary U_j to realize the operation $U_i(\theta_i)U_j(\theta_j)$, i.e., the application of $U_j(\theta_j)$ on the output of $U_i(\theta_i)$ at a particular gate parameter θ_j), as

$$\theta_{ij} = \theta_j, \tag{24}$$

such that arc s_{0j} is associated with $\theta_{0j} = \theta_j$.

Then a given state $x_{U_i(\theta_i)}$ of $\mathcal{X}$ associated with $U_i(\theta_i)$ is defined as

$$x_{U_i(\theta_i)} = v_{U_i} + a_{U_i(\theta_i)}, \tag{25}$$

where v_{U_i} is a label for unitary U_i in the environmental graph $\mathcal{G}_{\mathrm{QNN}_{QG}}$ (serves as an identifier in the computational structure of (25)), while parameter $a_{U_i(\theta_i)}$ is defined for a $U_i(\theta_i)$ as

$$a_{U_i(\theta_i)} = \sum_{h \in \Xi(i)} U_i(\theta_{hi}) x_{U_h(\theta_h)} + b_{U_i(\theta_i)}, \tag{26}$$

where $\Xi(i)$ refers to the parent set of v_{U_i}, $U_i(\theta_{hi})$ refers to the selection of θ_i for unitary U_i for a particular input from $U_h(\theta_h)$, while $b_{U_i(\theta_i)}$ is the bias relative to v_{U_i}.

Applying a $f_{\angle}$ topological ordering function on $\mathcal{G}_{\mathrm{QNN}_{QG}}$ yields an ordered graph structure $f_{\angle}(\mathcal{G}_{\mathrm{QNN}_{QG}})$ of the L unitaries. Thus, a given output $|Y\rangle$ of QNN_{QG} can be rewritten in a compact form as

$$|Y\rangle = U(\overrightarrow{\theta})x_0 = (U_L(\theta_L)U_{L-1}(\theta_{L-1}) \ldots U_1(\theta_1)x_0), \tag{27}$$

where the term $x_0 \in \mathcal{S}_{in}$ is associated with the input system as defined in (12).

A particular state $x_{U_l(\theta_l)}$, $l = 1, \ldots, L$ is evaluated in function of $x_{U_{l-1}(\theta_{l-1})}$ as

$$x_{U_l(\theta_l)} = U_l(\theta_l)x_{U_{l-1}(\theta_{l-1})}. \tag{28}$$

The environmental and ordered graphs of a gate-model quantum neural network are illustrated in Fig. 2. In Fig. 2(a) the $\mathcal{G}_{\mathrm{QNN}_{QG}}$ environmental graph of a QNN_{QG} is depicted, and the ordered graph $f_{\angle}(\mathcal{G}_{\mathrm{QNN}_{QG}})$ is shown in Fig. 2(b).

Computational model of gate-model quantum neural networks. **Theorem 1** *The computational model of a* QNN_{QG} *is a* $\mathcal{C}(QNN_{QG})$ *constraint machine with linear transition functions* $f_T(\mathrm{QNN}_{QG})$.

Proof. Let $\mathcal{G}(QNN_{QG}) = (V, S)$ be the environmental graph of a QNN_{QG}, and assume that the number of types of the vertexes is p. Then, the vertex set V can be expressed as a collection

$$V = \bigcup_{i=1}^{p} V_i, \tag{29}$$

where V_i identifies a set of vertexes, p is the total number of the V_i sets, such that $V_i \cap V_j = \varnothing$, if only $i \neq j$[48]. For a $v \in V_i$ vertex from set V_i, an $f_T : \mathbb{C}^{\dim_{in}} \rightarrow \mathbb{C}^{\dim_{out}}$ transition function[48] can be defined as

$$f_T : \mathcal{Z}_{V_i}^{|\Gamma(v)|} \times \mathcal{X}_{V_i} \rightarrow \mathcal{Z}_{V_i} : (\gamma_{\Gamma(v)}, x_v) \rightarrow f_T(\gamma_{\Gamma(v)}, x_v), \tag{30}$$

where $\mathcal{X}_{V_i}$ is the perceptual space $\mathcal{X}$ of V_i, $\mathcal{Z}_{V_i} \subset \mathbb{C}^{\dim(\mathcal{Z}_{V_i})}$; $\dim(\mathcal{Z}_{V_i})$ is the dimension of the space $\mathcal{Z}_{V_i}$; x_v is an element of $\mathcal{X}_{V_i}$; $x_v \in \mathcal{X}_{V_i}$ associated with a unitary $U_v(\theta_v)$; $\mathcal{Z}$ is the state space, $\mathcal{Z}_{V_i}$ is the state space of V_i, $\mathcal{Z}_{V_i} \subset \mathbb{C}^{\dim(\mathcal{Z}_{V_i})}$, $\dim(\mathcal{Z}_{V_i})$ is the dimension of the space $\mathcal{Z}_{V_i}$; $\Gamma(v)$ refers to the children set of v; $|\Gamma(v)|$ is the cardinality of set $\Gamma(v)$; $\gamma \in \mathcal{Z}$ is a state variable in the state space $\mathcal{Z}$ that serves as side information to process the v vertices of V in $\mathcal{G}(\mathrm{QNN}_{QG})$, while $\gamma_{\Gamma(v)} \in \mathcal{Z}_{V_i}^{|\Gamma(v)|} \subset \mathbb{C}^{|\Gamma(v)|}$ and $\gamma_{\Gamma(v)} = (\gamma_{\Gamma(v),1}, \cdots, \gamma_{\Gamma(v),|\Gamma(v)|})$, by theory[48,62]. Thus,

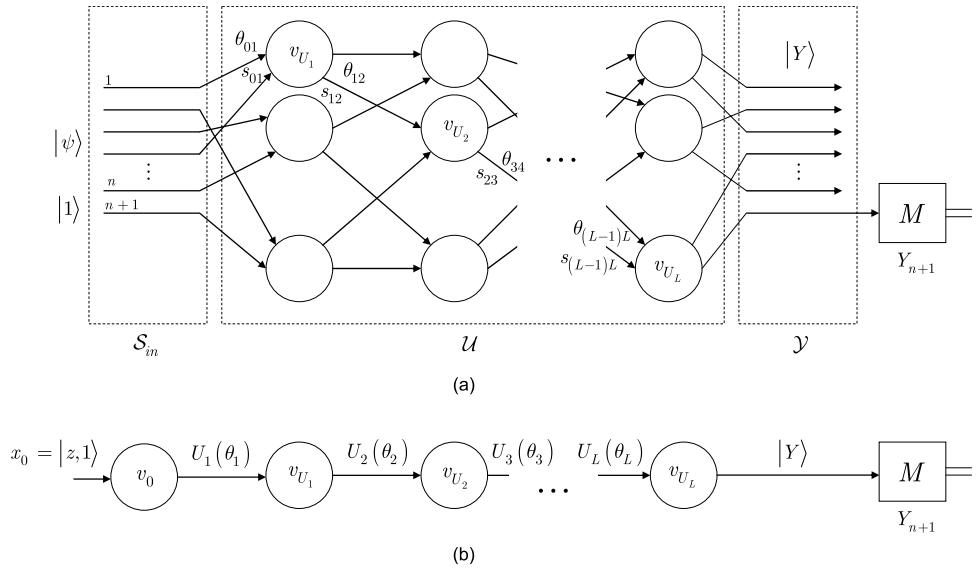

Figure 2. (**a**) The $\mathcal{G}_{\mathrm{RQNN}_{QG}}$ environmental graph of a QNN_{QG}, with L unitaries. The input state of the QNN_{QG} is $|\psi,1\rangle$. A unitary $U_i(\theta_i)$ is represented by a vertex v_{U_i} in $\mathcal{G}_{\mathrm{QNN}_{QG}}$. The vertices v_{U_i} and v_{U_j} of unitaries $U_i(\theta_i)$ and $U_j(\theta_j)$ are connected by directed arcs s_{ij}. The gate parameters $\theta_{ij}=\theta_j$ are associated with s_{ij}, while $\mathcal{S}_{in}$ is the input space, $\mathcal{U}$ is the space of L unitaries, and $\mathcal{Y}$ is the output space. Operator M is a measurement on the $(n+1)$-th state (readout quantum state), and Y_{n+1} is a Pauli operator measured on the readout state (classical links are not depicted) (**b**) The compacted ordered graph $f_{\angle}(\mathcal{G}_{\mathrm{QNN}_{QG}})$ of QNN_{QG}. The output is $|Y\rangle = U(\overrightarrow{\theta})x_0$, where $x_0 = |z,1\rangle$ and $U(\overrightarrow{\theta}) = \prod_{l=1}^{L} U_l(\theta_l)$ (classical links are not depicted).

the f_T transition function in (30) is a complex-valued function that maps an input pair (γ, x) from the space of $\mathcal{X} \times \mathcal{Z}$ to the state space $\mathcal{Z}$.

Similarly, for any V_i, an f_O: $\mathbb{C}^{\dim_{in}} \to \mathbb{C}^{\dim_{out}}$ output function[48] can be defined as

$$F_O\colon \mathcal{Z}_{V_i} \times \mathcal{X}_{V_i} \to \mathcal{Y}_{V_i}\colon (\gamma_v, x_v) \to F_O(\gamma_v, x_v), \tag{31}$$

where $\mathcal{Y}_{V_i}$ is the output space $\mathcal{Y}$, and γ_v is a state variable associated with v, $\gamma_v \in \mathcal{Z}_{V_i}$, such that $\gamma_v = \gamma_0$ if $\Gamma(v) = \varnothing$. The f_O output function in (31) is therefore a complex-valued function that maps an input pair (γ, x) from the space of $\mathcal{X} \times \mathcal{Z}$ to the output space $\mathcal{Y}$.

From (30) and (31), it follows that for any V_i, there exists the $\phi(V_i)$ associated function-pair as

$$\phi(V_i) = (f_T, F_O). \tag{32}$$

Let us specify the generalized functions of (30) and (31) for a QNN_{QG}.

Let $U(\overrightarrow{\theta})$ of QNN_{QG} be defined as given in (2). Since in QNN_{QG}, a given i-th unitary $U_i(\theta_i)$ acts on the output of the previous unitary $U_{i-1}(\theta_{i-1})$, the network contains no nonlinearities[14]. As a corollary, the state transition function $f_T(\mathrm{QNN}_{QG})$ in (30) is also linear for a QNN_{QG}.

Let $|\gamma_v\rangle$ be the quantum state associated with γ_v state variable of a given v. Then, the constraints on the transition function and output function of a QNN_{QG} can be evaluated as follows.

Let $f_T(\mathrm{QNN}_{QG})$ be the transition function of a QNN_{QG} defined for a given $v \in V$ of $\mathcal{G}(\mathrm{QNN}_{QG})$ via (30) as

$$f_T(\mathrm{QNN}_{QG})\colon (\gamma_{\Gamma(v)}, x_v) \to f_T(\gamma_{\Gamma(v)}, x_v). \tag{33}$$

The $F_O(\mathrm{QNN}_{QG})$ output function of a QNN_{QG} for a given v of $\mathcal{G}(\mathrm{QNN}_{QG})$ via (31) is

$$F_O(\mathrm{QNN}_{QG})\colon (\gamma_v, x_v) \to F_O(\gamma_v, x_v). \tag{34}$$

Since $f_T(\mathrm{QNN}_{QG})$ in (33) and $F_O(\mathrm{QNN}_{QG})$ in (34) correspond with the data flow computational scheme of a QNN_{QG} with linear transition functions, (33) and (34) represent an expression of the constraints of QNN_{QG}. These statements can be formulated in a compact form.

Let ζ_v be a constraint on $f_T(\mathrm{QNN}_{QG})$ of QNN_{QG} as

$$\zeta_v\colon |\gamma_v\rangle - f_T(QNN_{QG}) = 0. \tag{35}$$

Thus, the $f_T(\mathrm{QNN}_{QG})$ transition function is constrained as

$$f_T(\mathrm{QNN}_{QG}) = |\gamma_v\rangle. \tag{36}$$

With respect to the output function, let φ_v be a constraint on $F_O(\mathrm{QNN}_{QG})$ of QNN_{QG} as

$$\varphi_v\colon \wp_v \circ F_O(\mathrm{QNN}_{QG}) = 0, \tag{37}$$

where $\circ$ is the composition operator, such that $(f \circ g)(x) = f(g(x))$, $\wp_v$ is therefore another constraint as $\wp_v(F_O(\mathrm{QNN}_{QG})) = 0$.

Then let π_v be a compact constraint on $f_T(\mathrm{QNN}_{QG})$ and $F_O(\mathrm{QNN}_{QG})$ defined via constraints (35) and (37) as

$$\begin{aligned} \pi_v(f_T(\mathrm{QNN}_{QG}),\, F_O(\mathrm{QNN}_{QG})) & \\ &= \sum_{v \in V} (\zeta_v + \varphi_v) - 2|V| \\ &= \sum_{v \in V} ((|\gamma_v\rangle - (f_T(\mathrm{QNN}_{QG})) + \wp_v(F_O(QNN_{QG}))) - 2|V| \\ &\neq 0. \end{aligned} \tag{38}$$

Since it can be verified that a learning machine that enforces the constraint in (38), is in fact a constraint machine. As a corollary, the constraints (33) and (34), along with the compact constraint (38), define a $\mathcal{C}(\mathrm{QNN}_{QG})$ constraint machine for a QNN_{QG} with linear functions $f_T(\mathrm{QNN}_{QG})$ and $F_O(\mathrm{QNN}_{QG})$. ■

Diffusion machine. Let $\mathcal{C}$ be the constraint machine with linear transition function $f_T(\gamma_{\Gamma(v)}, x_v)$, and let $\S_v$ be a state variable such that $\forall\, v \in V$

$$\S_v - f_T(\gamma_{\Gamma(v)}, x_v) = 0, \tag{39}$$

and let $F_O(\gamma_v, x_v)$ be the output function of $\mathcal{C}$, such that $\forall v \in V$

$$c_v \circ F_O(\gamma_v, x_v) = 0, \tag{40}$$

where c_v is a constraint.

Then, the $\mathcal{C}$ constraint machine is a $\mathcal{D}$ diffusion machine[48], if only $\mathcal{C}$ enforces the constraint $C_{\mathcal{D}}$, as

$$C_{\mathcal{D}}\colon \sum_{v \in V} ((\S_v - f_T(\gamma_{\Gamma(v)}, x_v) = 0) + (c_v \circ F_O(\gamma_v, x_v) = 0)) - 2|V| = 0. \tag{41}$$

Computational model of recurrent gate-model quantum neural networks. **Theorem 2** *The computational model of an* RQNN_{QG} *is a* $\mathcal{D}(\mathrm{RQNN}_{QG})$ *diffusion machine with linear transition functions* $f_T(\mathrm{RQNN}_{QG})$.

Proof. Let $\mathcal{C}(\mathrm{RQNN}_{QG})$ be the constraint machine of RQNN_{QG} with linear transition function $f_T(\mathrm{RQNN}_{QG}) = f_T(\gamma_{\Gamma(v)}, x_v)$. Using the $\mathcal{G}_{\mathrm{RQNN}_{QG}}$ environmental graph, let Λ_v be a constraint on $f_T(\mathrm{RQNN}_{QG})$ of RQNN_{QG}, $v \in V$ as

$$\Lambda_v\colon |\gamma_v\rangle - f_T(\mathrm{RQNN}_{QG}) = 0, \tag{42}$$

where $|\gamma_v\rangle$ is the quantum state associated with γ_v state variable of a given v of RQNN_{QG}. With respect to the output function $F_O(\mathrm{RQNN}_{QG}) = F_O(\gamma_v, x_v)$ of RQNN_{QG}, let ω_v be a constraint on $F_O(\mathrm{RQNN}_{QG})$ of RQNN_{QG}, as

$$\omega_v\colon \Omega_v \circ F_O(\mathrm{RQNN}_{QG}) = 0, \tag{43}$$

where Ω_v is another constraint as $\Omega_v(F_O(\mathrm{RQNN}_{QG})) = 0$.

Since RQNN_{QG} is a recurrent network, for all $v \in V$ of $\mathcal{G}_{\mathrm{RQNN}_{QG}}$, a diffuse constraint $\lambda(Q(x))$ can be defined via constraints (42) and (43), as

$$\begin{aligned}\lambda(Q(x)) &= \sum_{v\in V}(\Lambda_v + \omega_v) - 2|V| \\ &= \sum_{v\in V}((|\gamma_v\rangle - (f_T(\mathrm{RQNN}_{QG})) + \Omega_v(F_O(\mathrm{RQNN}_{QG}))) - 2|V| \\ &= 0,\end{aligned} \tag{44}$$

where $x=(x_1, \ldots, x_{|V|})$, and $Q(x)=(Q(x_1), \ldots, Q(x_{|V|}))$ is a function that maps all vertexes of $\mathcal{G}_{\mathrm{RQNN}_{QG}}$. Therefore, in the presence of (44), the relation

$$\mathcal{C}(\mathrm{RQNN}_{QG}) = \mathcal{D}(\mathrm{RQNN}_{QG}), \tag{45}$$

follows for an RQNN_{QG}, where $\mathcal{D}(\mathrm{RQNN}_{QG})$ is the diffusion machine of RQNN_{QG}. It is because a constraint machine $\mathcal{C}(\mathrm{RQNN}_{QG})$ that satisfies (44) is, in fact, a diffusion machine $\mathcal{D}(\mathrm{RQNN}_{QG})$, see also (41).

In (42), the $f_T(\mathrm{RQNN}_{QG})$ state transition function can be defined for a $\mathcal{D}(\mathrm{RQNN}_{QG})$ via constraint (42) as

$$f_T(\mathrm{RQNN}_{QG}) = |\gamma_v\rangle. \tag{46}$$

Then, let H_t be a unit vector for a unitary $U_t(\theta_t)$, $t=1, \ldots, L-1$, defined as

$$H_t = x_t + iy_t, \tag{47}$$

where x_t and y_t are real values.

Then, let Z_{t+1} be defined via $U(\overrightarrow{\theta})$ and (47) as

$$Z_{t+1} = U(\overrightarrow{\theta})H_t + Ex_{t+1}, \tag{48}$$

where E is a basis vector matrix[60].

Then, by rewriting $U(\overrightarrow{\theta})$ as

$$U(\overrightarrow{\theta}) = \phi + i\varphi, \tag{49}$$

where ϕ, φ are real parameters, allows us to evaluate $U(\overrightarrow{\theta})H_t$ as

$$\begin{pmatrix}\mathrm{Re}(U(\overrightarrow{\theta})H_t)\\ \mathrm{Im}(U(\overrightarrow{\theta})H_t\end{pmatrix} = \begin{pmatrix}\phi & -\varphi\\ \varphi & \phi\end{pmatrix}\begin{pmatrix}x_t\\ y_t\end{pmatrix} \tag{50}$$

with

$$H_{t+1} = f_\sigma^{\mathrm{RQNN}_{QG}}(Z_{t+1}), \tag{51}$$

where H_{t+1} is normalized at unity, and function $f_\sigma^{\mathrm{RQNN}_{QG}}(\cdot)$ is defined as

$$f_\sigma^{\mathrm{RQNN}_{QG}}(Z) = \begin{cases} Z, & \text{if } |Z|_1 \geq 0\\ 0, & \text{if } |Z|_1 < 0\end{cases}, \tag{52}$$

where $|\cdot|_1$ is the $L1$-norm.

Since the RQNN_{QG} has linear transition function, (52) is also linear, and allows us to rewrite (52) via the environmental graph representation for a particular $(\gamma_{\Gamma(v)}, x_v)$, as

$$f_\sigma^{\mathrm{RQNN}_{QG}}(Z) = \begin{cases} f_T(\gamma_{\Gamma(v)}, x_v), & \text{if } |Z|_1 \geq 0\\ 0, & \text{if } |Z|_1 < 0\end{cases}, \tag{53}$$

where $f_T(\gamma_{\Gamma(v)}, x_v)$ is given in (50).

Thus, by setting $t=\nu$, the term H_t can be rewritten via (50) and (52) as

$$H_t = x_\nu = \mathrm{Re}(x_\nu) + i\ \mathrm{Im}(x_\nu). \tag{54}$$

Then, the $Y_t(\mathrm{RQNN}_{QG})$ output of RQNN_{QG} is evaluated as

$$Y_t(\mathrm{RQNN}_{QG}) = W\begin{pmatrix}\mathrm{Re}(x_\nu)\\ \mathrm{Im}(x_\nu)\end{pmatrix}, \tag{55}$$

where W is an output matrix[60].

Then let $|\Gamma(v)| = L$, therefore at a particular objective function $f(\theta)$ of the RQNN_{QG}, the derivative $\frac{df(\theta)}{dx_\nu}$ can be evaluated as

$$\begin{aligned}\frac{df(\theta)}{dx_\nu} &= \frac{df(\theta)}{dx_{|\Gamma(v)|}}\frac{dx_{|\Gamma(v)|}}{dx_\nu}\\ &= \frac{df(\theta)}{dx_{|\Gamma(v)|}}\prod_{k=\nu}^{|\Gamma(v)|-1}\frac{dx_{k+1}}{dx_k}\\ &= \frac{df(\theta)}{dx_{|\Gamma(v)|}}\prod_{k=\nu}^{|\Gamma(v)|-1}D_{k+1}(U(\overrightarrow{\theta}))^T,\end{aligned} \tag{56}$$

where

$$D_{k+1} = diag(Z_{k+1}) \tag{57}$$

is a Jacobian matrix[60]. For the norms the relation

$$\left\|\frac{df(\theta)}{dx_\nu}\right\| \le \left\|\frac{df(\theta)}{dx_{|\Gamma(v)|}}\right\|\prod_{k=\nu}^{|\Gamma(v)|-1}\|D_{k+1}(U(\overrightarrow{\theta}))^T\|, \tag{58}$$

holds, where

$$\|D_{k+1}(U(\overrightarrow{\theta}))^T\| = \|D_{k+1}\|. \tag{59}$$

The proof is concluded here. ■

Optimal Learning

Gate-model quantum neural network. **Theorem 3** *A supervised learning is an optimal learning for a* $\mathcal{C}(\mathrm{QNN}_{QG})$.

Proof. Let π_ν be the compact constraint on $f_T(\mathrm{QNN}_{QG})$ and $F_O(\mathrm{QNN}_{QG})$ of $\mathcal{C}(\mathrm{QNN}_{QG})$ from (38), and let A be a constraint matrix. Then, (38) can be reformulated as

$$\pi_v(f_T(\mathrm{QNN}_{QG}), F_O(\mathrm{QNN}_{QG})) = Af^*(x) - b(x) = 0. \tag{60}$$

where $b(x)$ is a smooth vector-valued function with compact support[48], $f^*: \mathcal{I} \to \mathbb{C}^n$,

$$f^*(x) = (f_T(\mathrm{QNN}_{QG}), F_O(\mathrm{QNN}_{QG}), x) \tag{61}$$

is the compact function subject to be determined such that

$$\forall\, x \in \mathcal{X}\colon \pi_v(v, f^*(x)) = 0. \tag{62}$$

The problem formulated via (60) can be rewritten as

$$Af^*(x) = b(x). \tag{63}$$

As follows, learning of functions $f_T(\mathrm{QNN}_{QG})$ and $F_O(\mathrm{QNN}_{QG})$ of $\mathcal{C}(\mathrm{QNN}_{QG})$ can be reduced to the determination of function $f^*(x)$, which problem is solvable via the Euler-Lagrange equations[48,63,64].

Then, let $\mathcal{S}_{L(\mathrm{QNN})}$ be a non-empty supervised learning set defined as a collection

$$\mathcal{S}_{L(\mathrm{QNN})}\colon \{x_\kappa, y_\kappa, \kappa \in \mathbb{N}_{\mathcal{X}}\}, \tag{64}$$

where (x_κ, y_κ), $y_\kappa = f^*(x_\kappa)$ is a supervised pair, and $|\mathcal{X}|$ is the cardinality of the perceptive space $\mathcal{X}$ associated with $\mathcal{S}_{L(\mathrm{QNN})}$.

Since $\mathcal{S}_{L(\mathrm{QNN})}$ is non-empty set, $f^*(x)$ can be evaluated by the Euler-Lagrange equations[48,63,64], as

$$f^*(x) = \frac{1}{\ell}\left(-A^T\lambda(x) - \frac{1}{|\mathcal{X}|}\sum_{\kappa=1}^{|\mathcal{X}|}(f^*(x) - y_\kappa)\Upsilon(x - x_\kappa)\right), \tag{65}$$

where A^T is the transpose of the constraint matrix A, and ℓ is a differential operator as

$$\ell = \sum_{\kappa=0}^{k}(-1)^\kappa c_\kappa \nabla^{2\kappa}, \tag{66}$$

where c_κ-s are constants, ∇^2 is a Laplacian operator such that $\nabla^2 f(x) = \sum_i \partial_i^2 f(x)$; while Y is as

$$\Upsilon(x - x_\kappa) = \ell\mathcal{G}(x, x_\kappa), \tag{67}$$

where $\mathcal{G}(\cdot)$ is the Green function of differential operator ℓ. Since function $\mathcal{G}(\cdot)$ is translation invariant, the relation

$$\mathcal{G}(x, x_\kappa) = \mathcal{G}(x - x_\kappa) \tag{68}$$

follows. Since the constraint that has to be satisfied over the perceptual space $\mathcal{X}$ is given in (62), the $\mathcal{L}$ Lagrangian can be defined as

$$\mathcal{L} = \langle Pf^*, Pf^*\rangle + \int_{\mathcal{X}} \lambda(x)\pi_v(x, f^*(x))dx, \tag{69}$$

where $\langle\cdot,\cdot\rangle$ is the inner product operator, while P is defined via (66) as

$$\ell = P^\dagger P, \tag{70}$$

where $P^\dagger$ is the adjoint of P, while $\lambda(x)$ is the Lagrange multiplier as

$$\lambda(x) = -(AA^T)^{-1}\left(\gamma\ell b(x) + \frac{1}{\mathcal{X}}\sum_{\kappa=1}^{|\mathcal{X}|}A(f^*(x) - y_\kappa)\Upsilon(x - x_\kappa)\right), \tag{71}$$

where

$$\gamma = \int_{\mathcal{X}} \mathcal{G}(x - x_\kappa), \tag{72}$$

and ℓb is as

$$\ell b(x) = -A\left(A^T\lambda(x) + \frac{1}{|\mathcal{X}|}\sum_{\kappa=1}^{|\mathcal{X}|}(f^*(x) - y_\kappa)\Upsilon(x - x_\kappa)\right). \tag{73}$$

Then, (65) can be rewritten using (71) and (73) as

$$f^*(x) = \frac{1}{\gamma\ell}\left(H(x) + \frac{1}{|\mathcal{X}|}\sum_{\kappa=1}^{|\mathcal{X}|}\Phi(y_\kappa - f^*(x))\Upsilon(x - x_\kappa)\right), \tag{74}$$

where $H(x)$ is as

$$H(x) = \gamma A^T(AA^T)^{-1}\ell b(x) \tag{75}$$

and Φ is as

$$\Phi = \mathbf{I}_n - A^T(AA^T)^{-1}A, \tag{76}$$

where $\mathbf{I}_n$ is an identity matrix.

Therefore, after some calculations, $f^*(x)$ can be expressed as

$$f^*(x) = \frac{1}{\gamma}\int_{\mathcal{X}} \mathcal{G}(z)H(x - z)dz + \sum_{\kappa=1}^{|\mathcal{X}|}\Phi\chi_\kappa\mathcal{G}(x - x_\kappa), \tag{77}$$

where χ_κ is as

$$\chi_\kappa = \frac{1}{|\mathcal{X}|}\frac{y_\kappa - f^*(x_\kappa)}{\gamma}. \tag{78}$$

Algorithm 1. Supervised learning for a $\mathcal{C}(\mathrm{QNN}_{QG})$.

Step 1. (Initialization.) Set $Y_{n+1}^{(r)}$, $r=1,\ldots,R$, where R is the number of total measurements applied for QNN_{QG}, and let $U(\vec{\theta})$ as given in (2).

Step 2. (Quantum evolution phase and parameter set.) Apply the unitary sequence $U(\vec{\theta})$ of QNN_{QG} realized via the L unitaries to produce output $|Y\rangle$. Let us assume that the node set V of $\mathscr{G}_{\mathrm{QNN}_{QG}}$ is topologically sorted via a topological ordering function $f_{\angle}(\mathscr{G}_{\mathrm{QNN}_{QG}})$.

Then, for a given $x_{U_i(\theta_i)}$ ((25)) set

$$V_{U_i(\theta_i)} = v_{U_i} + Q_{U_i(\theta_i)}, \tag{79}$$

where

$$Q_{U_i(\theta_i)} = \sum_{h\in\Xi(i)} \theta_{hi} V_{U_h(\theta_h)} + B_{v_{U_i}}, \tag{80}$$

where $\Xi(i)$ refers to the parent set of v_{U_i}, $B_{v_{U_i}}$ is a bias relative to v_{U_i}.

Step 3. (Error initialization.) Set the $\mathrm{P}^{(r)}(\mathscr{G}_{\mathrm{QNN}_{QG}})$ post-processing associated to the r-th measurement round on $\mathscr{G}_{\mathrm{QNN}_{QG}}$, as follows. For $i=1,\ldots,L$, set $W_{U_i(\theta_i)}$ as

$$W_{U_i(\theta_i)} = \sum_{h\in\Xi(i)} \theta_{hi} V_{U_h(\theta_h)}. \tag{81}$$

For $i=1,\ldots,L-1$, compute the error $\delta_{U_i(\theta_i)}$ associated to $U_i(\theta_i)$ as

$$\delta_{U_i(\theta_i)} = \frac{dW_{U_L(\theta_L)}}{dQ_{U_i(\theta_i)}}. \tag{82}$$

For $i=L$, set

$$\delta_{U_L(\theta_L)} = \frac{d\mathcal{L}(x_0,\tilde{l}(z))}{dQ_{U_L(\theta_L)}}. \tag{83}$$

Step 4. (Gate parameter updating.) Set a gate parameter modification vector $\vec{\Delta}\theta$ with an i-th element $\vec{\Delta}\theta_i$ as

$$\vec{\Delta}\theta_i = W_{U_i(\theta_i)}, \tag{84}$$

where $W_{U_i(\theta_i)}$ is given in (81). Update the gate parameters in a backpropagated manner from unitary $U_L(\theta_L)$ to $U_1(\theta_1)$, as follows. For $z=L,\ldots,1$:

If $\vec{\Delta}\theta_z = 1$, update θ_z as

$$\theta_z' = \theta_z. \tag{85}$$

If $\vec{\Delta}\theta_z \neq 1$, update θ_z as

$$\theta_z' = (\vec{\Delta}\theta_z)\theta_z. \tag{86}$$

Step 5. (Gradient computation). For $i=2,\ldots,L$, and $j\in\Xi(i)$, determine the gradient $g_{U_i(\theta_i),U_j(\theta_j)}$ between unitaries $U_i(\theta_i)$ and $U_j(\theta_j)$, as

$$g_{U_i(\theta_i),U_j(\theta_j)} = \delta'_{U_i(\theta_i)} W_{U_j(\theta_j)}, \tag{87}$$

where $\delta'_{U_i(\theta_i)}$ is the updated error evaluated as

$$\delta'_{U_i(\theta_i)} = (\vec{\Delta}\theta_i)\delta_{U_i(\theta_i)}. \tag{88}$$

Step 6. Apply steps 1-5, for $\forall r$ measurements.

The compact constraint of $\mathcal{C}(\mathrm{QNN}_{QG})$ determined via (77) is optimal, since (77) is the optimal solution of the Euler–Lagrange equations.

The proof is concluded here. ■

Lemma 1 *There exists a supervised learning for a $\mathcal{C}(\mathrm{QNN}_{QG})$ with complexity $\mathcal{O}(|S|)$, where $|S|$ is the number arcs (number of gate parameters) of $\mathcal{G}_{\mathrm{QNN}_{QG}}$.*

Proof. Let $\mathcal{G}_{\mathrm{QNN}_{QG}}$ be the environmental graph of QNN_{QG}, such that QNN_{QG} is characterized via $\overrightarrow{\theta}$ (see (3)).

The optimal supervised learning method of a $\mathcal{C}(\mathrm{QNN}_{QG})$ is derived through the utilization of the $\mathcal{G}_{\mathrm{QNN}_{QG}}$ environmental graph of QNN_{QG}, as follows.

The $\mathcal{A}_{\mathcal{C}(\mathrm{QNN}_{QG})}$ learning process of $\mathcal{C}(\mathrm{QNN}_{QG})$ in the $\mathcal{G}_{\mathrm{QNN}_{QG}}$ structure is given in Algorithm 1.

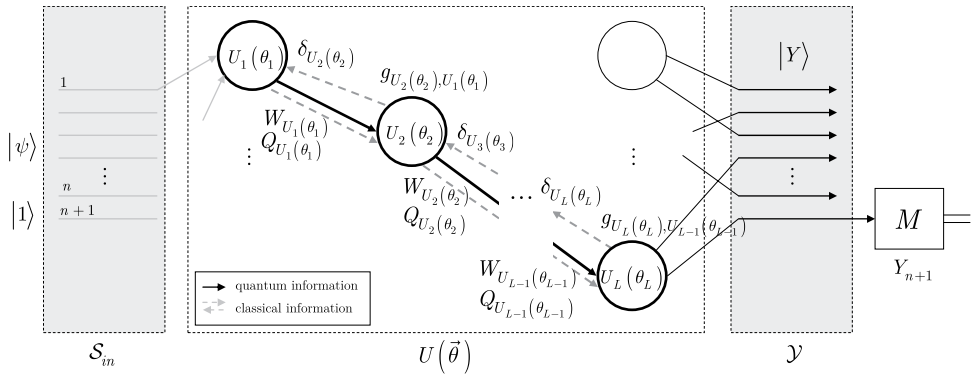

Figure 3. The learning method for a QNN_{QG}. The QNN_{QG} network realizes unitary $U(\vec{\theta})$ as a sequence of L unitaries, $U(\vec{\theta}) = U_L(\theta_L)U_{L-1}(\theta_{L-1})\dots U_1(\theta_1)$. The algorithm determines the gradient of the loss with respect to the θ gate parameter, at a particular loss function $\mathcal{L}(x_0, \tilde{I}(z))$. All quantum information propagates forward via quantum links (solid lines), classical side information can propagate arbitrarily (dashed lines).

The optimality of Algorithm 1 arises from the fact that in Step 4, the gradient computation involves all the gate parameters of the QNN_{QG}, and the gate parameter updating procedure has a computational complexity $\mathcal{O}(|S|)$. The QNN_{QG} complexity is yielded from the gate parameter updating mechanism that utilizes backpropagated classical side information for the learning method.

The proof is concluded here. ■

Description and method validation. The detailed steps and validation of Algorithm 1 are as follows.

In Step 1, the number R of measurement rounds is set.

Step 2 is the quantum evolution phase of QNN_{QG} that yields an output quantum system $|Y\rangle$ via forward propagation of quantum information through the unitary sequence $U(\vec{\theta})$ realized via the L unitaries. Then, a parameterization follows for each $x_{U_i(\theta_i)}$, and the terms $W_{U_i(\theta_i)}$ and $Q_{U_i(\theta_i)}$ are defined to characterize the θ_i angles of the $U_i(\theta_i)$ unitary operations in the QNN_{QG}.

In Step 3, side information initializations are made for the error computations. A given $W_{U_i(\theta_i)}$ is set as a cumulative quantity with respect to the parent set $\Xi \in i$ of unitary $U_i(\theta_i)$ in QNN_{QG}.

Note, that (80) and (81) represent side information, thus the gate parameter θ_{hi} is used to identify a particular unitary $U(\theta_{hi})$.

Let $\mathcal{G}'_{\text{QNN}_{QG}}$ be the the environmental graph of QNN_{QG} such that the directions of quantum links are reversed. It can be verified that for a $\mathcal{G}'_{\text{QNN}_{QG}}$, $\delta_{U_i(\theta_i)}$ from (82) can be rewritten as

$$\delta_{U_i(\theta_i)} = \sum_{h\in\Xi(i)} \frac{dW_{U_L(\theta_L)}}{dQ_{U_h(\theta_h)}} \frac{dQ_{U_h(\theta_h)}}{dW_{U_i(\theta_i)}} \frac{dW_{U_i(\theta_i)}}{dQ_{U_i(\theta_i)}} = Q_{U_i(\theta_i)} \sum_{h\in\Xi(i)} \theta_{hi}\delta_{U_h(\theta_h)}, \tag{89}$$

and $\delta_{U_L(\theta_L)}$ can be evaluated as given in (83)

$$\delta_{U_L(\theta_L)} = \frac{d\mathcal{L}(x_0, \tilde{I}(z))}{dQ_{U_L(\theta_L)}}, \tag{90}$$

while the term $\delta_{U_i(\theta_i)}W_{U_j(\theta_j)}$ for each $U_i(\theta_i)$ can be rewritten as

$$\delta_{U_i(\theta_i)}W_{U_j(\theta_j)} = \frac{d\mathcal{L}(x_0, \tilde{I}(z))}{d\theta_{ij}} = \frac{d\mathcal{L}(x_0, \tilde{I}(z))}{dQ_{U_i(\theta_i)}} \frac{dQ_{U_i(\theta_i)}}{d\theta_{ij}}. \tag{91}$$

Since (86) and (85) are defined via the non-reversed $\mathcal{G}_{\text{QNN}_{QG}}$, for a given unitary the Γ children set is used. The utilization of the Ξ parent set with reversed link directions in $\mathcal{G}'_{\text{QNN}_{QG}}$ (see (89), (90), (91)) is therefore analogous to the use of the Γ children set with non-reversed link directions in $\mathcal{G}_{\text{QNN}_{QG}}$. It is because classical side information is available in arbitrary directions in $\mathcal{G}_{\text{QNN}_{QG}}$.

First, we consider the situation, if $i = 1, \dots, L-1$, thus the error calculations are associated to unitaries $U_1(\theta_1)$, $\dots$, $U_{L-1}(\theta_{L-1})$, while the output unitary $U_L(\theta_L)$ is proposed for the $i = L$ case.

In $\mathcal{G}_{\text{QNN}_{QG}}$, the error quantity $\delta_{U_i(\theta_i)}$ associated to $U_i(\theta_i)$ is determined, where $W_{U_L(\theta_L)}$ is associated to the output unitary $U_L(\theta_L)$. Only forward steps are required to yield $W_{U_L(\theta_L)}$ and $Q_{U_L(\theta_L)}$. Then, utilizing the chain rule and using the children set $\Gamma(i)$ of a particular unitary $U_i(\theta_i)$, the term $dW_{U_L(\theta_L)}/dQ_{U_i(\theta_i)}$ in $\delta_{U_i(\theta_i)}$ can be rewritten as $\frac{dW_{U_L(\theta_L)}}{dQ_{U_i(\theta_i)}} = \sum_{h\in\Gamma(i)} \frac{dW_{U_L(\theta_L)}}{dQ_{U_h(\theta_h)}} \frac{dQ_{U_h(\theta_h)}}{dW_{U_i(\theta_i)}} \frac{dW_{U_i(\theta_i)}}{dQ_{U_i(\theta_i)}}$. In fact, this term equals to $Q_{U_i(\theta_i)} \sum_{h\in\Gamma(i)} \theta_{hi}\delta_{U_h(\theta_h)}$, where $\delta_{U_h(\theta_h)}$ is the error associated to a $U_h(\theta_h)$, such that $U_h(\theta_h)$ is a children unitary of $U_i(\theta_i)$. The $\delta_{U_h(\theta_h)}$ error quantity associated to a

children unitary $U_h(\theta_h)$ of $U_i(\theta_i)$ can also be determined in the same manner, that yields $\delta_{U_h(\theta_h)} = dW_{U_L(\theta_L)}/dQ_{U_h(\theta_h)}$. As follows, by utilizing side information in $\mathcal{G}_{\mathrm{QNN}_{QG}}$ allows us to determine $\delta_{U_i(\theta_i)}$ via the $\mathcal{L}(\cdot)$ loss function and the $\Gamma(i)$ children set of unitary $U_i(\theta_i)$, that yields the quantity given in (82).

The situation differs if the error computations are made with respect to the output system, thus for the L-th unitary $U_L(\theta_L)$. In this case, the utilization of the loss function $\mathcal{L}(x_0, \tilde{l}(z))$ allows us to use the simplified formula of $\delta_{U_L(\theta_L)} = d\mathcal{L}(x_0, \tilde{l}(z))/dQ_{U_L(\theta_L)}$, as given in (83). Taking the $\frac{d\mathcal{L}(x_0,\tilde{l}(z))}{d\theta_{ij}}$ derivative of the loss function $\mathcal{L}(x_0, \tilde{l}(z))$ with respect to the angle θ_{ij} yields $\frac{d\mathcal{L}(x_0,\tilde{l}(z))}{dQ_{U_i(\theta_i)}}\frac{dQ_{U_i(\theta_i)}}{d\theta_{ij}}$, that is, in fact equals to $\delta_{U_i(\theta_i)}W_{U_j(\theta_j)}$.

In Step 4, the quantities defined in the previous steps are utilized in the QNN_{QG} for the error calculations. The errors are evaluated and updated in a backpropagated manner from unitary $U_L(\theta_L)$ to $U_1(\theta_1)$. Since it requires only side information these steps can be achieved via a $\mathrm{P}(\mathcal{G}_{\mathrm{QNN}_{QG}})$ post-processing (along with Step 3). First, a gate parameter modification vector $\overrightarrow{\Delta}\theta$ is defined, such that its i-th element, $\overrightarrow{\Delta}\theta_i$, is associated with the modification of the θ_i gate parameter of an i-th unitary $U_i(\theta_i)$.

The i-th element $\overrightarrow{\Delta}\theta_i$ is initialized as $\overrightarrow{\Delta}\theta_i = W_{U_i(\theta_i)}$. If $\overrightarrow{\Delta}\theta_i$ equals to 1, then no modification is required in the θ_i gate parameter of $U_i(\theta_i)$. In this case, the $\delta_{U_i(\theta_i)}$ error quantity of $U_i(\theta_i)$ can be determined via a simple summation, using the children set of $U_i(\theta_i)$, as $\delta_{U_i(\theta_i)} = \sum_{j\in\Gamma(i)}\theta'_{ij}\delta_{U_j(\theta_j)}$, where $U_j(\theta_j)$ is a children of $U_i(\theta_i)$, as it is given in (85). On the other hand, if $\overrightarrow{\Delta}\theta_i \neq 1$, then the θ_i gate parameter of $U_i(\theta_i)$ requires a modification. In this case, summation $\sum_{j\in\Gamma(i)}\theta_{ij}\delta_{U_j(\theta_j)}$ has to be weighted by the actual $\overrightarrow{\Delta}\theta_i$ to yield $\delta_{U_i(\theta_i)}$. This situation is obtained in (86).

According to the update mechanism of (84–86), for $z = L-1, \ldots, 1$, the errors are updated via (88) as follows. At $z = L$ and $\overrightarrow{\Delta}\theta_z = 1$, $\delta_{U_z(\theta_z)}$ is as

$$\delta'_{U_z(\theta_z)} = \delta_{U_L(\theta_L)}. \tag{92}$$

while at $\overrightarrow{\Delta}\theta_z \neq 1$, $\delta_{U_z(\theta_z)}$ is updated as

$$\delta'_{U_z(\theta_z)} = (\overrightarrow{\Delta}\theta_z)\delta_{U_L(\theta_L)}. \tag{93}$$

For $z = L-1, \ldots, 1$, if $\overrightarrow{\Delta}\theta_z = 1$, then $\delta_{U_z(\theta_z)}$ is as

$$\delta'_{U_z(\theta_z)} = \delta_{U_z(\theta_z)} = \sum_{j\in\Gamma(z)}\theta_{zj}\delta_{U_j(\theta_j)}. \tag{94}$$

while, if $\overrightarrow{\Delta}\theta_z \neq 1$, then

$$\delta'_{U_z(\theta_z)} = (\overrightarrow{\Delta}\theta_z)\sum_{j\in\Gamma(z)}\theta_{zj}\delta_{U_j(\theta_j)} = \sum_{j\in\Gamma(z)}\theta'_{zj}\delta_{U_j(\theta_j)}. \tag{95}$$

In Step 5, for a given unitary $U_i(\theta_i)$, $i = 2, \ldots, L$ and for its parent $U_j(\theta_j)$, the $g_{U_i(\theta_i),U_j(\theta_j)}$ gradient is computed via the $\delta_{U_i(\theta_i)}$ error quantity derived from (85–86) for $U_i(\theta_i)$, and by the $W_{U_j(\theta_j)}$ quantity associated to parent $U_j(\theta_j)$. (For $U_1(\theta_1)$ the parent set $\Xi(1)$ is empty, thus $i > 1$.) The computation of $g_{U_i(\theta_i),U_j(\theta_j)}$ is performed for all $U_j(\theta_j)$ parents of $U_i(\theta_i)$, thus (87) is determined for $\forall j, j\in\Xi(i)$. By the chain rule,

$$\begin{aligned} g_{U_i(\theta_i),U_j(\theta_j)} = \delta'_{U_i(\theta_i)}W_{U_j(\theta_j)} &= \frac{dW_{U_L(\theta_L)}}{d\theta'_{ij}} \\ &= \frac{dW_{U_L(\theta_L)}}{dQ_{U_i(\theta_i)}}\frac{dQ_{U_i(\theta_i)}}{d\theta'_{ij}} \\ &= \frac{dW_{U_L(\theta_L)}}{dQ_{U_i(\theta_i)}}\frac{d(\sum_{h\in\Xi(i)}\theta_{hi}W_{U_h(\theta_h)})}{d\theta'_{ij}}. \end{aligned} \tag{96}$$

Since for $i = L$, $\delta_{U_L(\theta_L)}$ is as given in (83), the gradient can be rewritten via (91) as

$$g_{U_i(\theta_i),U_j(\theta_j)} = \frac{d\mathcal{L}(x_0, \tilde{l}(z))}{d\theta'_{ij}}. \tag{97}$$

Finally, Step 6 utilizes the number R of measurements to extend the results for all measurement rounds, $r = 1, \ldots, R$. Note that in each round a measurement operator is applied, for simplicity it is omitted from the description.

Since the algorithm requires no reversed quantum links, i.e. $\mathcal{G}'_{\mathrm{QNN}_{QG}}$ for the computations of (85–86), the gradient of the loss in (87) with respect to the gate parameter can be determined in an optimal way for QNN_{QG} networks, by the utilization of side information in $\mathcal{G}_{\mathrm{QNN}_{QG}}$.

The steps and quantities of the learning procedure (Algorithm 1) of a QNN_{QG} are illustrated in Fig. 3. The QNN_{QG} network realizes the unitary $U(\overrightarrow{\theta})$. The quantum information is propagated through quantum links

(solid lines) between the unitaries, while the auxiliary classical information is propagated via classical links in the network (dashed lines). An i-th node is represented via unitary $U_i(\theta_i)$.

For an i-th unitary, $U_i(\theta_i)$, parameters $W_{U_i(\theta_i)}$, $Q_{U_i(\theta_i)}$ and $\delta_{U_i(\theta_i)} = dW_{U_L(\theta_L)}/dQ_{U_i(\theta_i)}$ for $i < L$, are computed, where $W_{U_L(\theta_L)} = \sum_{j\in\Xi(L)}\theta_{Lj}V_{U_j(\theta_j)}$. For the output unitary, $\delta_{U_L(\theta_L)} = d\mathcal{L}(x_0, \tilde{l}(z))/dQ_{U_L(\theta_L)}$. Parameters $W_{U_i(\theta_i)}$ and $Q_{U_i(\theta_i)}$ are determined via forward propagation of side information, the $\delta_{U_i(\theta_i)}$ quantities are evaluated via backward propagation of side information. Finally, the gradients, $g_{U_i(\theta_i),U_j(\theta_j)} = \delta_{U_i(\theta_i)}W_{U_j(\theta_j)}$, are computed.

Recurrent gate-model quantum neural network. In classical neural networks, backpropagation[59–61] (backward propagation of errors) is a supervised learning method that allows to determine the gradients to learn the weights in the network. In this section, we show that for a recurrent gate-model QNN, a backpropagation method is optimal.

Theorem 4 *A backpropagation in $\mathcal{G}_{\mathrm{RQNN}_{QG}}$ is an optimal learning in the sense of gradient descent.*

Proof. In an RQNN_{QG}, the backward classical links provide feedback side information for the forward propagation of quantum information in multiple measurement rounds. The backpropagated side information is analogous to feedback loops, i.e, to recurrent cycles over time. The aim of the learning method is to optimize the gate parameters of the unitaries of the RQNN_{QG} quantum network via a supervised learning, using the side information available from the previous $k = 1, \ldots, r-1$ measurement rounds at a particular measurement round r.

Let $\mathcal{G}_{\mathrm{RQNN}_{QG}}$ be the environmental graph of RQNN_{QG}, and $f_T(\mathrm{RQNN}_{QG})$ be the transition function of an RQNN_{QG}. Then the γ_v constraint is defined via $\mathcal{G}_{\mathrm{RQNN}_{QG}}$ as

$$|\gamma_v\rangle = f_T(\mathrm{RQNN}_{QG}) = f_T(\gamma_{\Gamma(v)}, x_v), \tag{98}$$

while the constraint Ω_v on the output $F(\gamma_v, x_v)$ of RQNN_{QG} is defined via $\omega_v = 0$ as[48,61,62]

$$\omega_v\colon \Omega_v F(f_T(\mathrm{RQNN}_{QG}), x_v) = \Omega_v \circ F(f_T(\gamma_{\Gamma(v)}, x_v), x_v) = 0. \tag{99}$$

Utilizing the structure of the $\mathcal{G}_{\mathrm{RQNN}_{QG}}$ environmental graph allows us to define a modified version of the backpropagation through time algorithm[59] to the RQNN_{QG}.

The learning of $\mathcal{D}(\mathrm{RQNN}_{QG})$ with constraints (42), (43), and (44) is given in Algorithm 2, depicted as $\mathcal{A}_{\mathcal{D}(\mathrm{RQNN}_{QG})}$.

As a corollary, the training of $\mathcal{D}(\mathrm{RQNN}_{QG})$ can be reduced to a backpropagation method via the environmental graph of RQNN_{QG}. ■

Description and method validation. The detailed steps and validation of Algorithm 2 are as follows.

In Step 1, the number R of measurement rounds are set for RQNN_{QG}. For each measurement round initialization steps (100, 101) are set.

Step 2 provides the quantum evolution phase of RQNN_{QG}, and produces output quantum system $|Y_r\rangle$ (102) via forward propagation of quantum information through the unitary sequence $U(\overrightarrow{\theta_r})$ of the L unitaries.

Step 3 initializes the $\mathrm{P}^{(r)}(\mathrm{RQNN}_{QG})$ post-processing method via the definition of (105) for gradient computations. In (106), the quantity $\Phi_r = z^{(r)} + U(\overrightarrow{\theta_{r-1}}) + B_r$ connects the side information of the r-th measurement round with the side information of the $(r-1)$-th measurement round; and $U(\overrightarrow{\theta_{r-1}})$ is the unitary sequence of the $(r-1)$-th round, and B_r is a bias the current measurement round. The quantity $\xi_{r,k} = d\Phi_r/d\Phi_k$ in (107) utilizes the Φ_i quantities (see (106)) of the i-th measurement rounds, such that $i = k+1, \ldots, r$, where $k < r$.

Step 4 determines the g_r loss function gradient of the r-th measurement round. In (108), the g_r gradient is determined as $\sum_{k=1}^{r}\frac{\mathcal{L}(x_{0,r}, \tilde{l}(z^{(r)}))}{d\Phi_r}\frac{d\Phi_r}{d\Phi_k}\frac{d\Phi_k}{d\mathcal{S}(\overrightarrow{\theta_r})}$, that is, via the utilization of the side information of the $k = 1, \ldots, r$ measurement rounds at a particular r.

In Step 5, the gate parameters are updated via the gradient descent rule[59] by utilizing the gradients of the $k = 1, \ldots, r$ measurement rounds at a particular r. Since in (111) all the gate parameters of the L unitaries are updated by ω_r as given in (112), for a particular unitary $U_i(\theta_{r,i})$, the gate parameter is updated via $\overrightarrow{\alpha_r}$ (114) to $\theta_{r+1,i}$ as

Algorithm 2. *Optimal learning method for a* $\mathcal{D}(\mathrm{RQNN}_{QG})$.

Step 1. (Parameter initialization.) Set the number R of measurement rounds.
For an r-th measurement round, $r = 1, \ldots, R$, let

$$|\psi_r\rangle = |z^{(r)}\rangle \tag{100}$$

be the input quantum system of RQNN_{QG}, where $z^{(r)}$ is an n-length string, as

$$z^{(r)} = z_{r,1} z_{r,2} \ldots z_{r,n}. \tag{101}$$

Step 2. (Quantum evolution phase.) Evaluate the output system $|Y_r\rangle$ of RQNN_{QG} as

$$|Y_r\rangle = U(\vec{\theta}_r)|\psi_r\rangle|\varphi_r\rangle = U(\vec{\theta}_r)|\psi_r\rangle|1\rangle = U(\vec{\theta}_r)|z^{(r)}, 1\rangle, \tag{102}$$

where $\vec{\theta}_r$ is the gate-parameter vector associated to the L unitaries of RQNN_{QG} as

$$\vec{\theta}_r = \left(\theta_{r,1}, \ldots, \theta_{r,L-1}, \theta_{r,L}\right)^T, \tag{103}$$

and an r-th unitary sequence is as

$$U(\vec{\theta}_r) = U_L\left(\theta_{r,L}\right) U_{L-1}\left(\theta_{r,L-1}\right) \ldots U_1\left(\theta_{r,1}\right), \tag{104}$$

where $U_i\left(\theta_{r,i}\right)$ is the i-th unitary of $U(\vec{\theta}_r)$.
Step 3. (Post-processing initialization.) For a given r, initialize the $\mathrm{P}^{(r)}\left(\mathrm{RQNN}_{QG}\right)$ post-processing as follows. Define set $\mathscr{S}(\vec{\theta}_r)$ as

$$\mathscr{S}(\vec{\theta}_r) = \{\vec{\theta}_r, B_r\}, \tag{105}$$

where B_r is a bias.
Define Φ_r as

$$\Phi_r = z^{(r)} + U(\vec{\theta}_{r-1}) + B_r, \tag{106}$$

where $U(\vec{\theta}_{r-1})$ is the unitary sequence $U_L\left(\theta_{r-1,L}\right) U_{L-1}\left(\theta_{r-1,L-1}\right) \ldots U_1\left(\theta_{r-1,1}\right)$, of the $(r-1)$-th round.
Using (106), evaluate quantity $\xi_{r,k}$ as

$$\xi_{r,k} = \frac{d\Phi_r}{d\Phi_k} = \prod_{i=k+1}^{r} \frac{d\Phi_i}{d\Phi_{i-1}}, \tag{107}$$

where Φ_k belongs to the k-th measurement round, $k < r$.

Step 4. (Gradient computations). Using (107), compute the g_r loss function gradient of the r-th round as

$$g_r = \frac{\mathcal{L}\left(x_{0,r}, \tilde{l}(z^{(r)})\right)}{d\mathscr{S}(\vec{\theta}_r)} = \sum_{k=1}^{r} \frac{\mathcal{L}\left(x_{0,r}, \tilde{l}(z^{(r)})\right)}{d\Phi_r} \xi_{r,k} \frac{d\Phi_k}{d\mathscr{S}(\vec{\theta}_r)}, \tag{108}$$

where $\mathcal{L}\left(x_{0,r}, \tilde{l}(z^{(r)})\right)$ is the loss function of the r-th round,

$$\mathcal{L}\left(x_{0,r}, \tilde{l}(z^{(r)})\right) = 1 - l\left(z^{(r)}\right) \tilde{l}\left(z^{(r)}\right), \tag{109}$$

where $x_{0,r} = \left|z^{(r)}, 1\right\rangle$ identifies the input system of RQNN_{QG} in the r-th round, and $\tilde{l}\left(z^{(r)}\right)$ is as

$$\tilde{l}\left(z^{(r)}\right) = \langle z^{(r)}, 1\,|\,(U(\vec{\theta}_r))^{\dagger} Y_{n+1}^{(r)} U(\vec{\theta}_r)|z^{(r)}, 1\rangle, \tag{110}$$

where $\tilde{l}\left(z^{(r)}\right)$ is the predicted value of the binary label $l\left(z^{(r)}\right) \in \{-1, 1\}$ of string $z^{(r)}$, $Y_{n+1}^{(r)} \in \{-1, 1\}$ is a measured Pauli operator of r-th round, while $\frac{d\Phi_k}{d\mathscr{S}(\vec{\theta}_r)}$ is a partial derivative.
Step 5. (Gate parameter updates). If $r < R$, update gate parameter vector $\vec{\theta}_{r+1}$ via backpropagated side information as

$$\vec{\theta}_{r+1} = \vec{\theta}_r - \omega_r, \tag{111}$$

where ω_r is defined as

$$\omega_r = \frac{\lambda}{r} \sum_{k=1}^{r} g_k, \tag{112}$$

where g_r is the gradient (108) evaluated in the k-th measurement round,

$$g_k = \frac{\mathcal{L}\left(x_{0,k}, \tilde{l}(z^{(k)})\right)}{d\mathcal{S}\left(\vec{\theta}_k\right)}, \tag{113}$$

while λ is the learning rate.
Set the gate-parameter modification vector $\vec{\alpha}_r$ as

$$\vec{\alpha}_r = \left(\alpha_{r,1}, \ldots, \alpha_{r,L}\right)^T, \tag{114}$$

where $\alpha_{r,i}$ is associated to the modification of the gate parameter $\theta_{r,i}$ of the i-th unitary $U_i\left(\theta_{r,i}\right)$ as

$$\alpha_{r,i} = \omega_r. \tag{115}$$

Step 6. (Output gradient). Apply steps 1-5 for all r.
Output the G final gradient of the R rounds via the summation of the R gradients as

$$G = \sum_{r=1}^{R} g_r, \tag{116}$$

where g_r is given in (108).

$$\theta_{r+1,i} = \theta_{r,i} - \alpha_{r,i} = \theta_{r,i} - \omega_r. \tag{117}$$

Finally, Step 6 outputs the G final gradient of the total R measurement rounds in (116), as a summation of the g_r gradients (108) determined in the $r = 1, \ldots, R$ rounds.

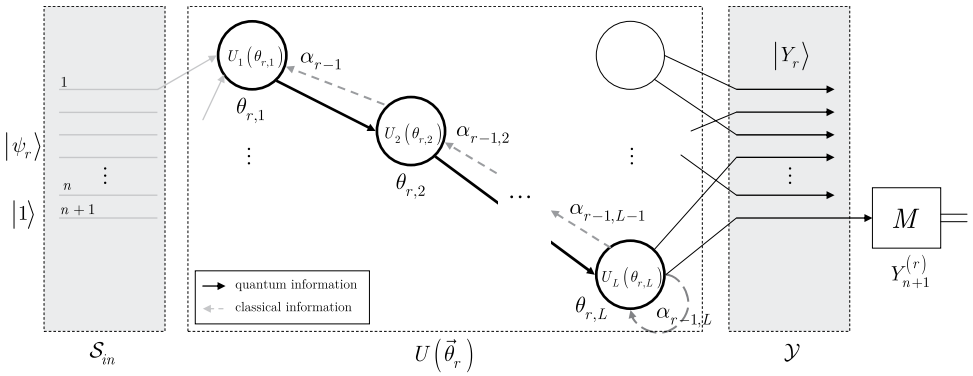

Figure 4. The learning method for an RQNN_{QG}. In an r-th measurement round, the RQNN_{QG} network realizes the unitary sequence $U(\overrightarrow{\theta_r})$, and side information is available about the previous $k=1,\ldots,r-1$ running sequences of the structure. Quantum information propagates only forward in the network via quantum links (solid lines), the $\alpha_{r-1,i}=\omega_{r-1}$ quantities are distributed via backpropagation of side information through the classical links (dashed lines). The $\theta_{r,j}$ gate parameter of an i-th unitary $U_i(\theta_{r,i})$ of $U(\overrightarrow{\theta_r})$ is set to $\theta_{r,i}=\theta_{r-1,i}-\alpha_{r-1,i}$, where $\theta_{r-1,i}$ is the gate parameter of the i-th unitary $U_i(\theta_{r-1,i})$ of the $U(\overrightarrow{\theta_{r-1}})$ unitary sequence.

The steps of the learning method of an RQNN_{QG} (Algorithm 2) are illustrated in Fig. 4. The $\overrightarrow{\theta_r}$ gate parameters of the unitaries of unitary sequence $U(\overrightarrow{\theta_r})$ are set as $\overrightarrow{\theta_r}=\overrightarrow{\theta_{r-1}}-\omega_{r-1}$, where $\overrightarrow{\theta_{r-1}}$ is the gate parameter vector associated to sequence $U(\overrightarrow{\theta_{r-1}})$, while $\alpha_{r-1,i}=\omega_{r-1}$ is the gate parameter modification coefficient, and $\omega_{r-1}=\frac{\lambda}{r-1}\sum_{k=1}^{r-1}\frac{\mathcal{L}(x_{0,k},\tilde{l}(z^{(k)}))}{d\mathcal{S}(\overrightarrow{\theta_k})}$.

Closed-form error evaluation. Lemma 2 *The δ quantity of the unitaries of a $\mathcal{D}(\mathrm{RQNN}_{QG})$ can be expressed in a closed form via the $\mathcal{G}_{\mathrm{RQNN}_{QG}}$ environmental graph of* RQNN_{QG}.

Proof. Let $\mathcal{G}_{\mathrm{RQNN}_{QG}}$ be the environmental graph of RQNN_{QG}, such that RQNN_{QG} is characterized via $\overrightarrow{\theta}$ (see (3)). Utilizing the structure $\mathcal{G}_{\mathrm{RQNN}_{QG}}$ of RQNN_{QG} allows us to express the square error in a closed form as follows.

Let Y and Z refer to output realizations $|Y\rangle$ and $|Z\rangle$ of RQNN_{QG}, $\mathcal{Y}\in Y, Z$, with an output set $\mathcal{Y}$, and let $\mathcal{L}(x_0,\tilde{l}(z))$ be the loss function. Then let $\mathbf{H}_{\mathrm{RQNN}_{QG}}$ be a Hessian matrix[48] of the RQNN_{QG} structure, with a generic coordinate $\hbar_{ij,lm}^{\mathrm{RQNN}_{QG}}$, as

$$\begin{aligned}
\hbar_{ij,lm}^{\mathrm{RQNN}_{QG}} &= \frac{d^2\mathcal{L}(x_0,\tilde{l}(z))}{d\theta_{ij}d\theta_{lm}}\\
&= \frac{d}{d\theta_{ij}}\sum_{Y\in\mathcal{Y}}\frac{d\mathcal{L}(x_0,\tilde{l}(z))}{dW_{U_Y(\theta_Y)}}\frac{dW_{U_Y(\theta_Y)}}{d\theta_{lm}}\\
&= \sum_{Y\in\mathcal{Y}}\sum_{Z\in\mathcal{Y}}\frac{d^2\mathcal{L}(x_0,\tilde{l}(z))}{dW_{U_Y(\theta_Y)}dW_{U_Z(\theta_Z)}}\delta^Z_{U_i(\theta_i)}\delta^Y_{U_l(\theta_l)}W_{U_j(\theta_j)}W_{U_m(\theta_m)}\\
&\quad+\sum_{Y\in\mathcal{Y}}\frac{d\mathcal{L}(x_0,\tilde{l}(z))}{dW_{U_Y(\theta_Y)}}\left(W_{U_m(\theta_m)}\frac{d\delta^Y_{U_l(\theta_l)}}{d\theta_{ij}}+\delta^Y_{U_l(\theta_l)}\frac{dW_{U_m(\theta_m)}}{d\theta_{ij}}\right),\\
&= \sum_{Y\in\mathcal{Y}}\sum_{Z\in\mathcal{Y}}\frac{d^2\mathcal{L}(x_0,\tilde{l}(z))}{dW_{U_Y(\theta_Y)}dW_{U_Z(\theta_Z)}}\delta^Z_{U_i(\theta_i)}\delta^Y_{U_l(\theta_l)}W_{U_j(\theta_j)}W_{U_m(\theta_m)}\\
&\quad+\sum_{Y\in\mathcal{Y}}\frac{d\mathcal{L}(x_0,\tilde{l}(z))}{dW_{U_Y(\theta_Y)}}\left(\left(\delta^Y_{U_l(\theta_l),U_i(\theta_i)}\right)^2W_{U_m(\theta_m)}W_{U_j(\theta_j)}+f_{i\angle m}\left(\delta^Y_{U_l(\theta_l)}\delta^m_{U_i(\theta_i)}W_{U_j(\theta_j)}\right)\right)
\end{aligned}\tag{118}$$

where $W_{U_i(\theta_i)}$ is given in (81), $f_{i\angle m}(\cdot)$ is a topological ordering function on $\mathcal{G}_{\mathrm{RQNN}_{QG}}$, indices Y and Q are associated with the output realizations $|Y\rangle$ and $|Q\rangle$, while $\left(\delta^Q_{U_l(\theta_l),U_i(\theta_i)}\right)^2$ is the square error between unitaries $U_l(\theta_l)$ and $U_i(\theta_i)$ at a particular output $|Q\rangle$ as

$$\begin{aligned}
\left(\delta^Q_{U_l(\theta_l),U_i(\theta_i)}\right)^2 &= \frac{d^2W_{U_Y(\theta_Y)}}{dQ_{U_l(\theta_l)}dQ_{U_i(\theta_i)}}=\frac{d\delta^Y_{U_l(\theta_l)}}{dQ_{U_l(\theta_l)}}\\
&= \frac{d\delta^i_{U_l(\theta_l)}}{dQ_{U_i(\theta_i)}}\sum_{j\in\Gamma(i)}\theta_{ji}\delta^Y_{U_l(\theta_l)}+Q_{U_i(\theta_i)}\sum_{j\in\Gamma(i)}\theta_{ji}\left(\delta^Y_{U_j(\theta_l),U_l(\theta_l)}\right)^2,
\end{aligned}\tag{119}$$

where $Q_{U_i(\theta_i)}$ is as in (80). Note that the relation $\left(\delta^{Q}_{U_i(\theta_i),U_i(\theta_i)}\right)^2 \neq 0$ in (119) holds if only there is an edge s_{il} between $v_{U_i} \in V$ and $v_{U_l(\theta_l)} \in V$ in the environmental graph $\mathcal{G}_{\mathrm{RQNN}_{QG}}$ of RQNN_{QG}. Thus,

$$\left(\delta^{Q}_{U_l(\theta_l),U_i(\theta_i)}\right)^2 = \begin{cases} \left(\delta^{Q}_{U_l(\theta_l),U_i(\theta_i)}\right)^2 = 0, & \text{if } s_{il} \notin S \\ \left(\delta^{Q}_{U_l(\theta_l),U_i(\theta_i)}\right)^2 \neq 0, & \text{if } s_{il} \in S \end{cases}. \tag{120}$$

Since $\mathcal{G}_{\mathrm{RQNN}_{QG}}$ contains all information for the computation of (119) and $\mathcal{D}(\mathrm{RQNN}_{QG})$ is defined through the structure of $\mathcal{G}_{\mathrm{RQNN}_{QG}}$, the proof is concluded here. ■

Conclusions

Gate-model QNNs allow an experimental implementation on near-term gate-model quantum computer architectures. Here we examined the problem of learning optimization of gate-model QNNs. We defined the constraint-based computational models of these quantum networks and proved the optimal learning methods. We revealed that the computational models are different for nonrecurrent and recurrent gate-model quantum networks. We proved that for nonrecurrent and recurrent gate-model QNNs, the optimal learning is a supervised learning. We showed that for a recurrent gate-model QNN, the learning can be reduced to backpropagation. The results are particularly useful for the training of QNNs on near-term quantum computers.

References

1. Preskill, J. Quantum Computing in the NISQ era and beyond. *Quantum* **2**, 79 (2018).
2. Harrow, A. W. & Montanaro, A. Quantum Computational Supremacy. *Nature* **549**, 203–209 (2017).
3. Aaronson, S. & Chen, L. Complexity-theoretic foundations of quantum supremacy experiments. *Proceedings of the 32nd Computational Complexity Conference*, CCC '17, 22:1–22:67, (2017).
4. Biamonte, J. *et al*. Quantum Machine Learning. *Nature* **549**, 195–202 (2017).
5. LeCun, Y., Bengio, Y. & Hinton, G. Deep Learning. *Nature* **521**, 436–444 (2014).
6. Goodfellow, I., Bengio, Y. & Courville, A. *Deep Learning*. MIT Press. Cambridge, MA (2016).
7. Debnath, S. *et al*. Demonstration of a small programmable quantum computer with atomic qubits. *Nature* **536**, 63–66 (2016).
8. Monz, T. *et al*. Realization of a scalable Shor algorithm. *Science* **351**, 1068–1070 (2016).
9. Barends, R. *et al*. Superconducting quantum circuits at the surface code threshold for fault tolerance. *Nature* **508**, 500–503 (2014).
10. Kielpinski, D., Monroe, C. & Wineland, D. J. Architecture for a large-scale ion-trap quantum computer. *Nature* **417**, 709–711 (2002).
11. Ofek, N. *et al*. Extending the lifetime of a quantum bit with error correction in superconducting circuits. *Nature* **536**, 441–445 (2016).
12. IBM. *A new way of thinking: The IBM quantum experience*. URL, http://www.research.ibm.com/quantum (2017).
13. Brandao, F. G. S. L., Broughton, M., Farhi, E., Gutmann, S. & Neven, H. For Fixed Control Parameters the Quantum Approximate Optimization Algorithm's Objective Function Value Concentrates for Typical Instances. *arXiv* **1812**, 04170 (2018).
14. Farhi, E. & Neven, H. Classification with Quantum Neural Networks on Near Term Processors. *arXiv* **1802**, 06002v1 (2018).
15. Farhi, E., Goldstone, J., Gutmann, S. & Neven, H. Quantum Algorithms for Fixed Qubit Architectures. *arXiv* **1703**, 06199v1 (2017).
16. Farhi, E., Goldstone, J. & Gutmann, S. A Quantum Approximate Optimization Algorithm. *arXiv* **1411**, 4028 (2014).
17. Farhi, E., Goldstone, J. & Gutmann, S. A Quantum Approximate Optimization Algorithm Applied to a Bounded Occurrence Constraint Problem. *arXiv* **1412**, 6062 (2014).
18. Lloyd, S. The Universe as Quantum Computer, *A Computable Universe: Understanding and exploring Nature as computation*, H. Zenil ed., World Scientific, Singapore, 2012, *arXiv:1312.4455v1* (2013).
19. Lloyd, S., Mohseni, M. & Rebentrost, P. Quantum algorithms for supervised and unsupervised machine learning. *arXiv* **1307**, 0411v2 (2013).
20. Lloyd, S., Mohseni, M. & Rebentrost, P. Quantum principal component analysis. *Nature Physics* **10**, 631 (2014).
21. Rebentrost, P., Mohseni, M. & Lloyd, S. Quantum Support Vector Machine for Big Data Classification. *Phys. Rev. Lett.* **113** (2014).
22. Lloyd, S., Garnerone, S. & Zanardi, P. Quantum algorithms for topological and geometric analysis of data. *Nat. Commun.* **7**, arXiv:1408. 3106 (2016).
23. Schuld, M., Sinayskiy, I. & Petruccione, F. An introduction to quantum machine learning. *Contemporary Physics* **56**, pp. 172–185. *arXiv: 1409.3097* (2015).
24. Imre, S. & Gyongyosi, L. *Advanced Quantum Communications - An Engineering Approach*. Wiley-IEEE Press (New Jersey, USA) (2012).
25. Dorozhinsky, V. I. & Pavlovsky, O. V. Artificial Quantum Neural Network: quantum neurons, logical elements and tests of convolutional nets, *arXiv:1806.09664* (2018).
26. Torrontegui, E. & Garcia-Ripoll, J. J. Universal quantum perceptron as efficient unitary approximators, *arXiv:1801.00934* (2018).
27. Lloyd, S. *et al*. Infrastructure for the quantum Internet. *ACM SIGCOMM Computer Communication Review* **34**, 9–20 (2004).
28. Gyongyosi, L., Imre, S. & Nguyen, H. V. A Survey on Quantum Channel Capacities. *IEEE Communications Surveys and Tutorials* **99**, 1, https://doi.org/10.1109/COMST.2017.2786748 (2018).
29. Van Meter, R. *Quantum Networking*, John Wiley and Sons Ltd, ISBN 1118648927, 9781118648926 (2014).
30. Gyongyosi, L. & Imre, S. Multilayer Optimization for the Quantum Internet. *Scientific Reports*, Nature, https://doi.org/10.1038/s41598-018-30957-x, (2018).
31. Gyongyosi, L. & Imre, S. Entanglement Availability Differentiation Service for the Quantum Internet. *Scientific Reports*, Nature, https://doi.org/10.1038/s41598-018-28801-3, https://www.nature.com/articles/s41598-018-28801-3 (2018).
32. Gyongyosi, L. & Imre, S. Entanglement-Gradient Routing for Quantum Networks. *Scientific Reports*, Nature, https://doi.org/10.1038/s41598-017-14394-w, https://www.nature.com/articles/s41598-017-14394-w, (2017).
33. Gyongyosi, L. & Imre, S. Decentralized Base-Graph Routing for the Quantum Internet, *Physical Review A*, American Physical Society, https://doi.org/10.1103/PhysRevA.98.022310, https://link.aps.org/doi/10.1103/PhysRevA.98.022310 (2018).
34. Pirandola, S., Laurenza, R., Ottaviani, C. & Banchi, L. Fundamental limits of repeaterless quantum communications, *Nature Communications*, 15043, https://doi.org/10.1038/ncomms15043 (2017).
35. Pirandola, S. *et al*. Theory of channel simulation and bounds for private communication. *Quantum Sci. Technol.* **3**, 035009 (2018).
36. Laurenza, R. & Pirandola, S. General bounds for sender-receiver capacities in multipoint quantum communications. *Phys. Rev. A* **96**, 032318 (2017).
37. Pirandola, S. Capacities of repeater-assisted quantum communications. *arXiv* **1601**, 00966 (2016).
38. Pirandola, S. End-to-end capacities of a quantum communication network. *Commun. Phys.* **2**, 51 (2019).

39. Cacciapuoti, A. S. *et al*. Quantum Internet: Networking Challenges in Distributed Quantum Computing. *arXiv* **1810**, 08421 (2018).
40. Shor, P. W. Scheme for reducing decoherence in quantum computer memory. *Phys. Rev. A* **52**, R2493–R2496 (1995).
41. Petz, D. *Quantum Information Theory and Quantum Statistics*, Springer-Verlag, Heidelberg, Hiv: 6. (2008).
42. Bacsardi, L. On the Way to Quantum-Based Satellite Communication. *IEEE Comm. Mag.* **51**(08), 50–55 (2013).
43. Gyongyosi, L. & Imre, S. A Survey on Quantum Computing Technology, *Computer Science Review*, Elsevier, https://doi.org/10.1016/j.cosrev.2018.11.002, ISSN: 1574-0137 (2018).
44. Wiebe, N., Kapoor, A. & Svore, K. M. Quantum Deep Learning. *arXiv* **1412**, 3489 (2015).
45. Wan, K. H. *et al*. Quantum generalisation of feedforward neural networks. *npj Quantum Information 3, 36 arXiv* **1612**, 01045 (2017).
46. Cao, Y., Giacomo Guerreschi, G. & Aspuru-Guzik, A. Quantum Neuron: an elementary building block for machine learning on quantum computers. *arXiv: 1711.11240* (2017).
47. Lloyd, S. & Weedbrook, C. Quantum generative adversarial learning. *Phys. Rev. Lett., 121, arXiv* **1804**, 09139 (2018).
48. Gori, M. *Machine Learning: A Constraint-Based Approach*, ISBN: 978-0-08-100659-7, Elsevier (2018).
49. Hyland, S. L. & Ratsch, G. Learning Unitary Operators with Help From u(n). *arXiv* **1607**, 04903 (2016).
50. Dunjko, V. *et al*. Super-polynomial and exponential improvements for quantum-enhanced reinforcement learning. *arXiv: 1710.11160* (2017).
51. Romero, J. *et al*. Strategies for quantum computing molecular energies using the unitary coupled cluster ansatz. *arXiv: 1701.02691* (2017).
52. Riste, D. *et al*. Demonstration of quantum advantage in machine learning. *arXiv* **1512**, 06069 (2015).
53. Yoo, S. *et al*. A quantum speedup in machine learning: finding an N-bit Boolean function for a classification. *New Journal of Physics* **16**(10), 103014 (2014).
54. Farhi, E. & Harrow, A. W. Quantum Supremacy through the Quantum Approximate Optimization Algorithm. *arXiv* **1602**, 07674 (2016).
55. Crooks, G. E. Performance of the Quantum Approximate Optimization Algorithm on the Maximum Cut Problem. *arXiv* **1811**, 08419 (2018).
56. Gyongyosi, L. & Imre, S. Dense Quantum Measurement Theory. *Scientific Reports*, Nature, https://doi.org/10.1038/s41598-019-43250-2 (2019).
57. Farhi, E., Kimmel, S. & Temme, K. A Quantum Version of Schoning's Algorithm Applied to Quantum 2-SAT. *arXiv* **1603**, 06985 (2016).
58. Schoning, T. A probabilistic algorithm for *k*-SAT and constraint satisfaction problems. *Foundations of Computer Science*, 1999. 40th Annual Symposium on, pages 410–414. IEEE (1999).
59. Salehinejad, H., Sankar, S., Barfett, J., Colak, E. & Valaee, S. Recent Advances in Recurrent Neural Networks. *arXiv* **1801**, 01078v3 (2018).
60. Arjovsky, M., Shah, A. & Bengio, Y. Unitary Evolution Recurrent Neural Networks. *arXiv: 1511.06464* (2015).
61. Goller, C. & Kchler, A. Learning task-dependent distributed representations by backpropagation through structure. *Proc. of the ICNN-96*, pp. 347–352, Bochum, Germany, IEEE (1996).
62. Baldan, P., Corradini, A. & Konig, B. Unfolding Graph Transformation Systems: Theory and Applications to Verification, In: Degano P., De Nicola R., Meseguer J. (eds) *Concurrency, Graphs and Models. Lecture Notes in Computer Science*, vol 5065. Springer, Berlin, Heidelberg (2008).
63. Roubicek, T. *Calculus of variations. Mathematical Tools for Physicists*. (Ed. Grinfeld, M.) J. Wiley, Weinheim, ISBN 978-3-527-41188-7, pp. 551–588 (2014).
64. Binmore, K. & Davies, J. *Calculus Concepts and Methods*. Cambridge University Press. p. 190. ISBN 978-0-521-77541-0. OCLC 717598615. (2007).

Acknowledgements

The research reported in this paper has been supported by the National Research, Development and Innovation Fund (TUDFO/51757/2019-ITM, Thematic Excellence Program). This work was partially supported by the National Research Development and Innovation Office of Hungary (Project No. 2017-1.2.1-NKP-2017-00001), by the Hungarian Scientific Research Fund - OTKA K-112125 and in part by the BME Artificial Intelligence FIKP grant of EMMI (BME FIKP-MI/SC).

Author Contributions

L.GY. designed the protocol and wrote the manuscript. L.GY. and S.I. analyzed the results. All authors reviewed the manuscript.

Additional Information

Supplementary information accompanies this paper at https://doi.org/10.1038/s41598-019-48892-w.

Competing Interests: The authors declare no competing interests.

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