Variational quantum reinforcement learning via evolutionary optimization

IOP Publishing Mach. Learn.: Sci. Technol. 3 (2022) 015025 https://doi.org/10.1088/2632-2153/ac4559

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
8 September 2021

REVISED
7 December 2021

ACCEPTED FOR PUBLICATION
21 December 2021

PUBLISHED
15 February 2022

PAPER

Variational quantum reinforcement learning via evolutionary optimization

Samuel Yen-Chi Chen[1,*], **Chih-Min Huang**[2], **Chia-Wei Hsing**[2], **Hsi-Sheng Goan**[2,3,4,5] and **Ying-Jer Kao**[2,3,4]

[1] Computational Science Initiative, Brookhaven National Laboratory, Upton, NY 11973, United States of America
[2] Department of Physics, National Taiwan University, Taipei 10617, Taiwan
[3] Center for Theoretical Physics, National Taiwan University, Taipei 10617, Taiwan
[4] Center for Quantum Science and Engineering, National Taiwan University, Taipei 10617, Taiwan
[5] Physics Division, National Center for Theoretical Science, Taipei 10617, Taiwan
[*] Author to whom any correspondence should be addressed.

E-mail: ychen@bnl.gov

Keywords: quantum machine learning, artificial intelligence, variational quantum circuits, quantum neural networks, reinforcement learning, evolutionary optimization

Abstract

Recent advances in classical reinforcement learning (RL) and quantum computation point to a promising direction for performing RL on a quantum computer. However, potential applications in quantum RL are limited by the number of qubits available in modern quantum devices. Here, we present two frameworks for deep quantum RL tasks using gradient-free evolutionary optimization. First, we apply the amplitude encoding scheme to the Cart-Pole problem, where we demonstrate the quantum advantage of parameter saving using amplitude encoding. Second, we propose a hybrid framework where the quantum RL agents are equipped with a hybrid tensor network-variational quantum circuit (TN-VQC) architecture to handle inputs of dimensions exceeding the number of qubits. This allows us to perform quantum RL in the MiniGrid environment with 147-dimensional inputs. The hybrid TN-VQC architecture provides a natural way to perform efficient compression of the input dimension, enabling further quantum RL applications on noisy intermediate-scale quantum devices.

1. Introduction

Recently available quantum computers (QCs) [1–3], which can potentially have significant speedup over classical computers on certain problems [4–7], show great promise in the application of machine learning tasks. However, these so-called noisy intermediate-scale quantum (NISQ) devices [8] lack quantum error correcting capabilities [5, 9, 10] and cannot perform quantum algorithms with a large number of qubits and a deep circuit. These limitations make the development of near-term quantum algorithms highly non-trivial.

Numerous efforts have been made to utilize these NISQ resources, and one of the notable achievements is *variational quantum algorithms* (VQAs) [11, 12]. In such a framework, certain parts of a given computational task that can leverage the strength of quantum physics will be put on a QC, while the rest remains on a classical computer. The outputs from the QC will be channeled into the classical computer, and a predefined algorithm will determine how to adjust the parameters of the quantum circuit on the QC.

VQAs have been applied in various machine learning tasks, such as classification and function approximation [13]. It has been shown that VQA may have extraordinary expressive power [14] and can be employed to solve complex machine learning (ML) problems. Classical reinforcement learning (RL) [15] has demonstrated remarkable ability in achieving better than human performance on video games [16–19], and the game of Go [20, 21]. Recently, it has also been applied to quantum physics research, such as quantum control [22–24], quantum error correction [25–27] and quantum experiment design [28]. How to replicate the successes of classical RL by incorporating VQA into the RL model, in particular, with environments of large dimensions, remains an open question.

Existing quantum RL schemes rely solely on gradient-based methods to optimize the policy and/or value functions [29–33]. On the other hand, evolutionary optimization has been shown to reach similar or even superior performance compared to gradient-based methods on certain difficult RL problems [34]. To the best of our knowledge, the application of evolutionary algorithms to quantum RL optimization has not been studied extensively. Another hurdle of quantum RL is that current quantum devices have only a few qubits, thus preventing the potential use of cases of environments with large dimensions. Here, we present an evolutionary and gradient-free method to optimize the quantum circuit parameters for RL agents. We show that this method can successfully train a quantum deep RL model to achieve a state-of-the-art result on the Cart-Pole problem with only a few parameters, demonstrating a potential quantum advantage. In addition, we demonstrate that the evolutionary method can be used to optimize models combining tensor networks (TNs) and variational quantum circuits (VQCs) in an end-to-end manner, opening up more opportunities for the application of quantum RL with NISQ devices.

In this work, we present an evolutionary deep quantum RL framework to demonstrate potential quantum advantage. We contribute the following:

- Demonstrate the quantum advantage of parameter saving via amplitude encoding. In the Cart-Pole environment, we successfully use the amplitude encoding to encode a 4D input vector into a two-qubit system.
- Demonstrate the capabilities of the hybrid TN-VQC architecture in quantum RL scenarios. The hybrid architecture can efficiently compress the large dimensional input into a small representation that can be processed with an NISQ device.

The paper is organized as follows. In section 2, we introduce the basics of RL. In section 3, we describe the testing environments used in this work. In section 4, we introduce the basics of VQCs. Section 5 describes the quantum circuit architecture for the Cart-Pole problem. Section 6 introduces TN methods and describes the hybrid TN-VQC architecture for the MiniGrid problem. Section 7 explains the evolutionary method used to optimize quantum circuit parameters. The performance of the proposed models is shown in section 8, followed by further discussions in section 9. Finally, we conclude our work in section 10.

2. Reinforcement learning

RL is a machine learning paradigm where a given goal is to be achieved through an *agent* interacting with an *environment,* $\mathcal{E}$, over a sequence of discrete time steps [15]. At each time step, t, the agent observes a *state,* s_t, and subsequently selects an *action,* a_t, from a set of possible actions $\mathcal{A}$ according to its current *policy,* π. The policy is a mapping from a certain state, s_t, to the probability of selecting an action from $\mathcal{A}$. After performing the action, a_t, the agent receives a scalar *reward*, r_t, and the state of the next time step, s_{t+1}. For episodic tasks, the process proceeds over a number of time steps until the agent reaches the terminal state. An *episode* includes all the states that the agent experienced throughout the aforementioned process, from a randomly selected initial state to the terminal state. Along each state, s_t, during the training process, the agent's overall goal is to maximize the expected return, which is quantified by the value function at state s under policy π, $V^\pi(s) = \mathbb{E}[R_t|s_t = s]$, where $R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$ is the *return*, together with the total discounted reward from time step t. The discount factor $\gamma \in (0, 1]$ allows the investigator to control the influence of future rewards on the agent's decision making. A large discount rate γ forces the agent to take into account the distant future, whereas a small γ allows the agent to focus more on immediate rewards and ignore future rewards beyond a few time steps. The value function can be expressed as $V^\pi(s) = \sum_{a\in\mathcal{A}} Q^\pi(s,a)\pi(a|s)$, where the *action-value function* or *Q-value function* $Q^\pi(s,a) = \mathbb{E}[R_t|s_t = s, a]$ is the expected return of choosing an action $a \in \mathcal{A}$ in state s according to the policy π. Selecting the best policy among all possible policies yields the maximal action-value, given by the optimal action-value function $Q^*(s,a) = \max_\pi Q^\pi(s,a)$, which in turn produces the maximal expected return. In this work, we construct a hybrid quantum–classical model consisting of TNs and VQCs to approximate the functionality of the Q-value function. The idea is that given a state s, the model will output an array of values corresponding to the relative goodness of each action. Then, the action with the highest values will be selected.

3. Testing environments

Here, we briefly describe the two testing environments in our experiments: Cart-Pole and MiniGrid [35].

3.1. Cart-Pole

Cart-Pole is a common testing environment for benchmarking simple RL models and has been a standard example in the OpenAI Gym [36] (see figure 1). In this environment, a pole is attached by a fixed joint to a

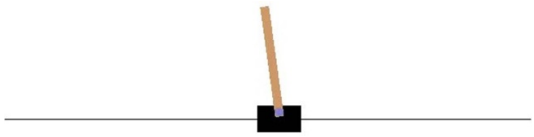

Figure 1. Cart-Pole environment from OpenAI Gym.

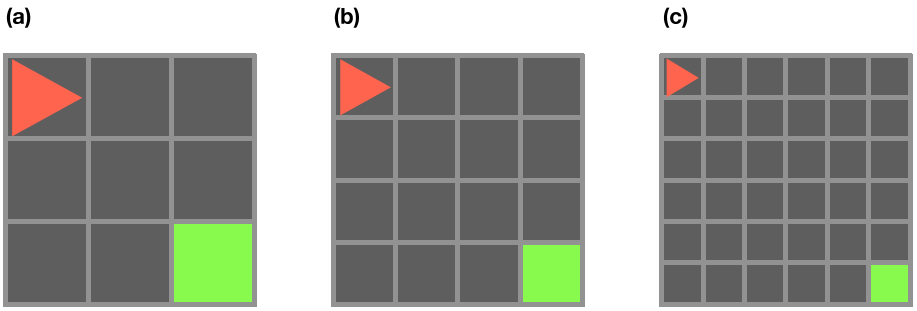

Figure 2. Empty environment from MiniGrid. We consider three environments from MiniGrid in this work. In each environment, there are also walls which span 1 unit on each side (not shown in the figure). (a), (b) and (c) represent the `MiniGrid-Empty-5x5-v0`, `MiniGrid-Empty-6x6-v0` and `MiniGrid-Empty-8x8-v0` environments, respectively.

cart moving horizontally along a frictionless track. The pendulum initially stays upright, and the goal is to keep it as close to the initial state as possible by pushing the cart leftwards and rightwards. The RL agent learns to output the appropriate action according to the observation it receives at each time step.

The Cart-Pole environment mapping is as follows:

- Observation: a 4D vector s_t comprising values of the cart position, cart velocity, pole angle, and pole velocity at the tip.
- Action: there are two actions $+1$ and -1 in the action space, corresponding to pushing the cart rightwards and leftwards, respectively.
- Reward: a reward of $+1$ is given for every time step where the pole remains close to being upright. An episode terminates if the pole is angled over 15 degrees from vertical or the cart moves more than 2.4 units away from the center.

3.2. MiniGrid

MiniGrid [35] is a more complicated environment with a much larger observation input for the RL agent. In this environment, the RL agent receives a $7 \times 7 \times 3 = 147$ dimensional vector from observation and accordingly has to determine the action from the action space $\mathcal{A}$, which contains a total of six possibilities. Note that the 147-dimensional vector is a compact and efficient encoding of the environment, not the actual pixels. As shown in figure 2, the agent (shown in a red triangle) is expected to find the shortest path from the starting point to the goal (shown in green).

The MiniGrid environment mapping is as follows:

- Observation:a 147 dimensional vector s_t.
- Action: there are six actions $0, \ldots, 5$ in $\mathcal{A}$, each corresponding to the following:
  - Turn left
  - Turn right
  - Move forwards
  - Pick up an object
  - Drop the object being carried
  - Toggle (open doors, interact with objects)
- Reward: a reward of 1 is given when the agent reaches the goal. A penalty is subtracted from the reward according to the formula:

$$1 - 0.9 \times (\textit{number of steps}/\textit{max steps allowed}). \tag{1}$$

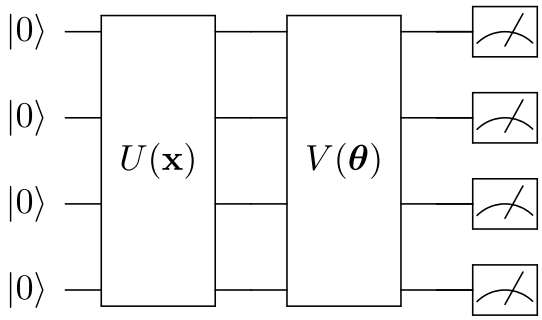

Figure 3. Generic architecture for VQCs. $U(\mathbf{x})$ is the quantum circuit block for encoding the (classical) input data $\mathbf{x}$ into a quantum state and $V(\boldsymbol{\theta})$ is the variational block with learnable parameters $\boldsymbol{\theta}$ to be optimized via gradient-based or gradient-free methods. Quantum measurements are performed over some or all of the qubits.

The *max steps allowed* is defined to be $4 \times n \times n$ where n is the grid size [35]. In the present study, we consider $n = 5, 6, 8$. This reward scheme is challenging since it is *sparse*, i.e. the agent will not receive any reward along the steps until it reaches the goal and therefore most of the actions elicit no immediate response from the environment.

4. Variational quantum circuits

VQCs are quantum circuits with parameters tunable via iterative optimizations, which are done on a classical computer with either gradient-based [37, 38] or gradient-free algorithms [39]. The architecture of a generic VQC is illustrated in figure 3. The $U(\mathbf{x})$ block serves as the state preparation part, which encodes the classical data $\mathbf{x}$ into the circuit quantum states and is not subject to optimization, whereas the $V(\boldsymbol{\theta})$ block represents the variational part containing *trainable* parameters $\boldsymbol{\theta}$, which in this study are optimized through evolutionary methods. The output information is obtained by measuring a subset or all of the qubits and thereby retrieve a classical bit string.

One of the notable advantages of these circuits is that they are resistant to quantum noise [40–42], potentially favorable for NISQ devices. In the field of quantum machine learning (QML), applications of VQCs to standard machine learning tasks have achieved various degrees of success. Prominent examples include function approximation [13, 43–45], classification [13, 14, 46–63], generative modeling [64–68], deep RL [29–33, 69–72], sequence modeling [43, 73–76], speech recognition [77], metric and embedding learning [78, 79], transfer learning [50, 80] and federated learning [77, 81, 82]. Furthermore, it is shown that the VQCs have more expressive power than classical neural networks [83–85]. Whether VQCs can perform better than their classical counterparts with an equal or fewer number of parameters is therefore an interesting subject to explore.

5. Quantum architecture for the Cart-Pole problem

In this problem, the observation input is 4D, which can be readily encoded into a quantum circuit. The quantum circuit for the Cart-Pole experiment is shown in figure 4. In this two-qubit architecture, we load the 4D observation input with *amplitude encoding*. $U(\boldsymbol{x})$ represents the quantum routine for amplitude encoding, of which possible implementations are described in [86, 87]. Here, we repeat the variational circuit block (shown in a grouped box) four times to increase the number of parameters and thereby the expressive power. In the measurement part, since there are only two possible actions in the Cart-Pole problem, we simply evaluate the Z expectation values of the two qubits individually. The measurement output, i.e. the expectation values, are then further processed by adding a classical *bias* of the same dimension (2).

5.1. Amplitude encoding

As the observation space of the Cart-Pole environment is continuous, it is impossible to use the computational basis encoding (which is for discrete space, as used in the previous work [29]) to encode the input state. For this task, we employ the *amplitude encoding* method to transform the observation into the amplitude of a quantum state. *Amplitude encoding* is a method to encode a vector $(\alpha_0, \ldots, \alpha_{2^N-1})$ into an N-qubit quantum state $|\Psi\rangle = \alpha_0|00\cdots0\rangle + \cdots + \alpha_{2^N-1}|11\cdots1\rangle$ where the α_i are real numbers and the vector $(\alpha_0, \ldots, \alpha_{2^N-1})$ is normalized. A potential advantage is that for an m-dimensional vector, it requires only $\log_2(m)$ qubits to encode the data. The details of this operation are described in appendix A.2. The full quantum circuit simulation is performed with the package `PennyLane`.

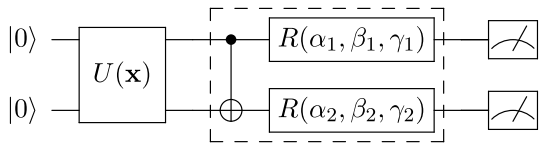

Figure 4. Quantum circuit architecture for the Cart-Pole problem. $U(\mathbf{x})$ is the quantum routine for amplitude encoding and the grouped box is the variational circuit block with tunable parameters α_i, β_i, γ_i. In the evolutionary quantum RL architecture for the Cart-Pole problem, the grouped box repeats four times. Total number of parameters is $2 \times 3 \times 4 + 2 = 26$ where the extra two parameters are the bias added after the measurement.

5.2. Action selection

The selection of the next action is similar to that used in the quantum deep Q-learning in [29]. Specifically, the output from the quantum circuit after the classical post-processing of this two-qubit system is a two-tuple $[a, b]$. If a is the maximum between the two values, then the corresponding action is -1. On the other hand, if the maximum is b, then the action is $+1$.

6. Hybrid TN-VQC architecture for the MiniGrid problem

One of the key challenges in the NISQ era is that quantum-computing machines are typically equipped with a limited number of qubits and can only execute quantum algorithms with a small circuit depth. To process data, with input dimensions exceeding the number of available qubits, it is necessary to apply certain kinds of dimensional reduction techniques to first compress the input data. For example, in [50], the authors applied a classical pre-trained convolution neural network to reduce the input dimension and then used a small VQC model to classify the images. However, the pre-trained model is already sufficiently powerful, and it is not clear whether the VQC plays a critical role in the whole process.

On the other hand, in [55], the authors explored the possibilities of using a TN for feature extraction and training the TN-VQC hybrid model in an end-to-end fashion. It has been shown that this hybrid TN-VQC architecture succeeds in classification tasks. However, to the best of our knowledge, the potential of this architecture has not yet been explored in RL schemes with complex environments. In the MiniGrid, the observation is a 147-dimensional vector and is impossible to be directly processed on current NISQ devices. Here, we propose a hybrid TN-VQC agent architecture (see figure 5) so that an efficient dimensional reduction can be achieved.

6.1. Tensor network

TN is a technique originally developed in the field of quantum many-body physics [88–94] for efficiently expressing the quantum wave function $|\Psi\rangle$. MPS, among others, is a type of 1D TN that decomposes a large tensor into a series of matrices. A general N-qubit quantum state can be written as,

$$|\Psi\rangle = \sum_{i_1}\sum_{i_2}\cdots\sum_{i_N} T_{i_1 i_2 \cdots i_N} |i_1\rangle \otimes |i_2\rangle \otimes \cdots \otimes |i_N\rangle \tag{2}$$

where $T_{i_1 i_2 \cdots i_N}$ is the amplitude of each basis state $|i_1\rangle \otimes |i_2\rangle \otimes \cdots \otimes |i_N\rangle$. As the number of entries of $T_{i_1 i_2 \cdots i_N}$ grows exponentially with N, it is extremely difficult to store and process quantum states on a classical computer when the number of qubits becomes large. However, with MPS, an efficient representation that approximately decomposes $T_{i_1 i_2 \cdots i_N}$ into a product of matrices [95]:

$$T_{i_1 i_2 \cdots i_N} = \sum_{\alpha_1}\sum_{\alpha_2}\cdots\sum_{\alpha_N} A^1_{i_1 \alpha_1} A^2_{\alpha_1 i_2 \alpha_2} A^3_{\alpha_2 i_2 \alpha_3} \cdots A^N_{\alpha_{N-1} i_N}, \tag{3}$$

where the matrices A are indexed from 1 to N and α_j represent the virtual indices, one can largely reduce the space where $|\Psi\rangle$ resides. Each *virtual index* α_j has a dimension m called *bond dimension* and serves as a tunable hyperparameter in the MPS approximation. It is known that for a sufficiently large m, an MPS can represent any tensor [96]. In machine learning applications, the bond dimension m is typically used to tune the number of trainable parameters and thereby the expressive power of the MPS. Figure 6 shows the illustration of tensors and MPS. We refer to [97] for an in-depth introduction to TNs.

Since the pioneering work of [98], great effort has been made to apply TN in the field of machine learning. TN-based methods have been utilized for applications, such as classification [98–104], generative modeling [105–107] and sequence modeling [108]. It has also been shown that TN-based architectures have deep connections to the building of QML models [109]. Specifically, we show that it is possible to encode a

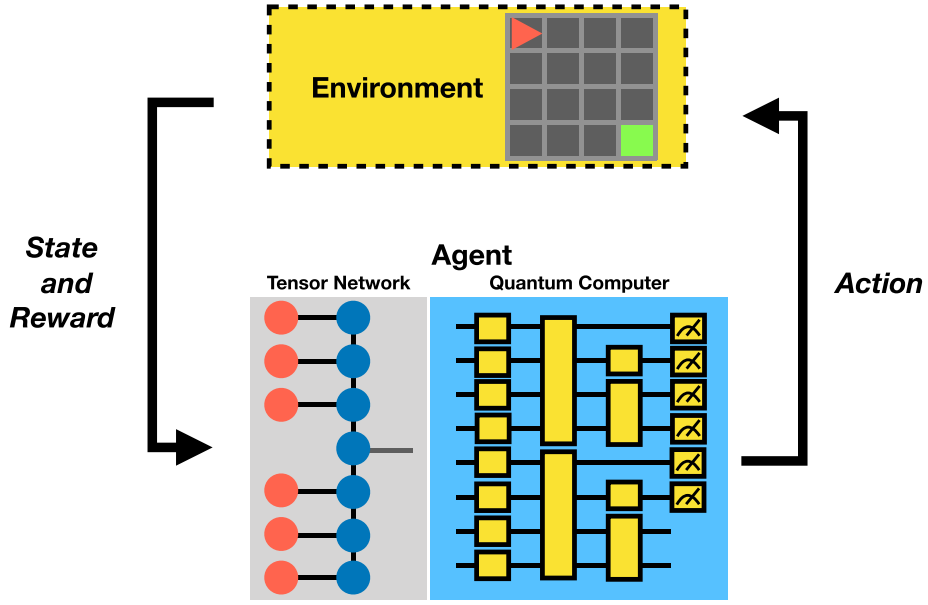

Figure 5. Hybrid TN-VQC agent for the MiniGrid environment. In this framework, the observation or state, which is of a dimension exceeding the capabilities of available QCs, is first compressed into a vector via the trainable matrix product states (MPS). Compressed representation is then encoded into the VQC for further training. Action is chosen based on the output of the VQC. In this work, we train this TN-VQC model in an end-to-end fashion.

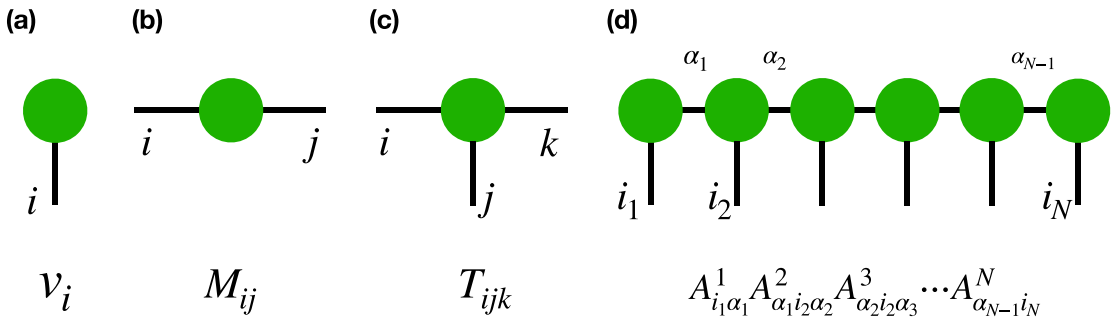

Figure 6. Graphical notation of tensors and TNs. Graphical tensor notation of (a) a vector, (b) a matrix, (c) a rank-3 tensor and (d) an MPS. We follow the Einstein convention that repeated indices, represented as closed legs in the diagram, are summed over.

quantum-inspired TN architecture, such as MPS, into a quantum circuit with single- and two-qubit gates [110].

6.2. MPS operation

In the TN-VQC architecture in this study, we use an MPS-based feature extractor as the TN part to reduce the input dimension. For an MPS to process an input vector $\mathbf{v}$, a *feature map* $\Phi(\mathbf{v})$ is needed. The most general form is:

$$\mathbf{v} \to |\Phi(\mathbf{v})\rangle = \phi(v_1) \otimes \phi(v_2) \otimes \cdots \otimes \phi(v_N), \tag{4}$$

where each ϕ is a d-dimensional feature map, mapping each v_j into a d-dimensional vector. The value d is known as the *local dimension*. In this work, we choose $d = 2$ and the feature map $\phi(v_j)$ to be:

$$\phi(v_j) = \begin{bmatrix} 1 - v_j \\ v_j \end{bmatrix}. \tag{5}$$

The input vector $\mathbf{v}$, a state/observation perceived by the agent, is therefore encoded into a tensor product state in the following way:

$$\mathbf{v} \to |\Phi(\mathbf{v})\rangle = \begin{bmatrix} 1 - v_1 \\ v_1 \end{bmatrix} \otimes \begin{bmatrix} 1 - v_2 \\ v_2 \end{bmatrix} \otimes \cdots \otimes \begin{bmatrix} 1 - v_N \\ v_N \end{bmatrix}, \tag{6}$$

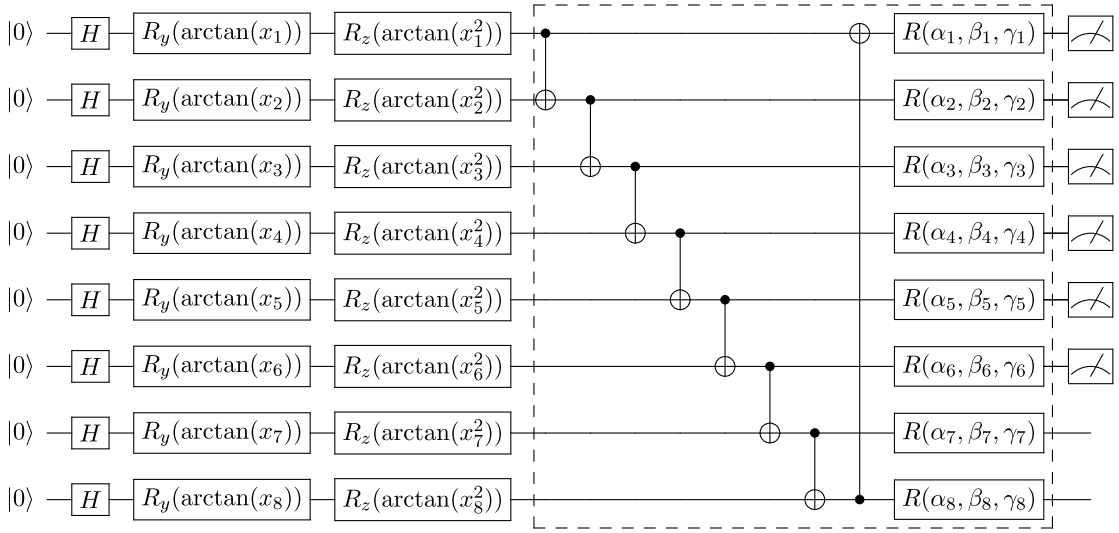

Figure 7. VQC architecture for the action selection. VQC component in our hybrid TN-VQC architecture includes three parts. First one is the *encoding* part, comprising Hadamard gates H and single-qubit rotation gates $R_y(\arctan(x_i))$ and $R_z(\arctan(x_i^2))$. Hadamard gate H serves to generate an unbiased initial state as described in equation (A1), and the two rotation gates are for state preparation, where the input parameters x_i are the compressed representation generated by the MPS component. Second one is the *variational* part, comprising CNOT gates, which entangle the quantum states of the qubits, and general rotation gates $R(\alpha,\beta,\gamma)$, where the parameters α_i, β_i and γ_i are to be optimized via the learning process. Final one is the *measurement* part, which gives the Pauli-Z expectation values through multiple runs of the quantum circuit, where the quantum measurements are performed on the first k qubits, with k being the number of actions. Retrieved values (logits) subsequently undergo classical processing, such as the softmax function to yield the *probability* of each possible action.

which is then contracted with the trainable MPS and becomes a vector:

$$f(\mathbf{v}) = \sum_{i_1}\sum_{i_2}\cdots\sum_{i_N} T_{i_1 i_2 \ldots i_N}\phi(v_1)_{i_1}\,\phi(v_2)_{i_2}\cdots\phi(v_N)_{i_N}\,, \tag{7}$$

where $i_1, i_2, \ldots, i_N$ are in $\{0,1\}$ and $T_{i_1 i_2 \ldots i_N}$ is defined as in equation (3) but with an additional rank-3 tensor in the middle with an open leg representing the 8D output, i.e. the compressed representation, as can be seen schematically in figure 5. In figure 5, the feature-mapped input and the trainable MPS are shown in red and blue circles (nodes), respectively. Since the observation input in MiniGrid is a 147-dimensional vector, there are in total 147 input nodes and $(147+1)$ MPS nodes.

6.3. VQC processing

For the VQC part, we adopt the *variational encoding* method to encode our compressed representations into quantum states. The basic idea of this encoding method is that the values to be encoded can be used as quantum rotation (e.g. R_y and R_z) angles directly or after some simple transformation. This method is generally easy to implement on an NISQ device. The details of this operation are described in appendix A.1.

The encoded state is then processed with the *variational* part with optimizable parameters, as shown in the dashed box in figure 7. The box can be repeated multiple times to increase the number of parameters and thus the expressive power of the model. For the MiniGrid problem, we use only one block since the performance does not increase significantly with more than one variational block. In the final part of the model, the Pauli-Z expectation values are retrieved through multiple runs (shots) of the circuit. Only the first six qubits are measured. These values are then processed with a softmax function to evaluate the probabilities of each action.

6.4. Action selection

The selection of the next action is similar to that used in the quantum deep Q-learning in [29]. Specifically, the output from the quantum circuit after the classical post-processing of this eight-qubit system is a six-tuple $[a,b,c,d,e,f]$. If $a/b/c/d/e/f$ is the maximum among the six values, then the corresponding action is $0/1/2/3/4/5$.

7. Quantum circuit evolution

Here, we elucidate our quantum circuit evolution algorithm inspired by [34]. The essential concept of this approach is to first generate a population of agents with random parameters and then make them evolve

through a number of generations with a certain mutation rate. In each generation, the fittest agents will be selected to produce the next generation. The details of each step are explained below. See appendix B for the pseudocode of the whole quantum circuit evolution algorithm.

7.1. Initialization

We first initialize the population $\mathcal{P}$ of N agents with each of them given randomly generated initial parameters $\boldsymbol{\theta}$, which are sampled from the normal distribution $\mathcal{N}(0, I)$ and multiplied by a factor of 0.01. The multiplication factor 0.01 serves to set the parameters around zero, thereby rendering the training process more stable.

7.2. Running the agents

For each generation, all of the agents' fitness is evaluated as follows. Each agent plays the game for R_1 times and the average score, which represents the fitness, is calculated by $S_i^{avg} = \frac{1}{R_1}\sum_{r=1}^{R_1} S_{i,r}$ where $S_{i,r}$ is the score of the rth trial obtained by the ith agent. The score is simply the sum of rewards within an episode. Taking into account the fitness of all the agents, we then select the top T agents according to their average scores (fitness) S_i^{avg}. The resulting group is called the *parents* and used to generate the next generation.

7.3. Mutation and the next generation

The N children of the next generation are generated via two separate procedures. In the first part, we generate a group of $N-1$ children, each of which is a single agent randomly selected from the parent group and slightly mutated. Specifically, the parameter vector $\boldsymbol{\theta}$ of this parent agent undergoes the following *mutation* operation: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \sigma\epsilon$, where σ is the *mutation power* and ε the Gaussian noise sampled from the normal distribution $\mathcal{N}(0, I)$. This is distinct from the commonly used gradient-based methods in that the optimization direction is randomly chosen, a feature that can potentially provide the advantage of circumventing the local optima and efficiently optimizing the parameters in an environment with sparse rewards [34]. The second part is to find the *elite*, the Nth child, which is not mutated. To make the selection process more robust against noise, we make each agent from the parent group play the game R_2 times and obtain the average scores E_j^{avg}. The top-1 agent, i.e. the one with the highest average score E_j^{avg}, is selected as the elite child.

8. Experiments and results

We first demonstrate the quantum advantage of VQC with amplitude encoding in the standard benchmark Cart-Pole environment. Then, we show the capabilities of our hybrid TN-VQC architecture in processing a larger dimensional input state in a MiniGrid environment. The procedure of evolutionary optimization is the same for both experiments.

8.1. Cart-Pole

In this experiment, we set the number of generations to be 1700, the population size $N = 500$, the truncation selection number $T = 5$, the mutation power $\sigma = 0.02$, the number of repetitions (for evaluating all the agents) $R_1 = 3$ and the number of repetitions (for evaluating the parents) $R_2 = 5$. The simulation results of this experiment are shown in figure 8. It can be seen that after about 250 generations, the average score of the top-5 agents is steadily above 400, and after around 1300 generations, the top-5 agents all converge to the optimal policy and reach a score of 500, which corresponds to the maximum number of steps allowed in the environment.

A notable achievement is that we use only 26 parameters to reach the optimal result, which is less than those required in typical classical neural networks by at least one order of magnitude. Empowered by amplitude encoding as well as the nature of VQC, we significantly reduce the number of parameters in this specific problem. It is thus highly desirable to explore the feasibility of applying such an encoding method to other quantum RL problems, which could bring about a quantum advantage as a way of reducing the model complexity as quantified by the number of parameters to as small as poly$(\log n)$, in contrast to the poly(n) parameters typically required in standard neural networks where n is the dimension of the input vector [86].

8.2. MiniGrid

Here, we consider three configurations, as shown in figure 2. The observation is a 147-dimensional vector, and there are six possible actions, as described in section 3.2. In this experiment, we employ hybrid TN-VQC architecture combining MPS and VQC. The MPS feature extractor receives the 147-dimensional input state from the environment and outputs an 8D vector to be encoded into the VQC. Empowered by the TN method, we successfully compress the large input vector into a small vector representation favorable for

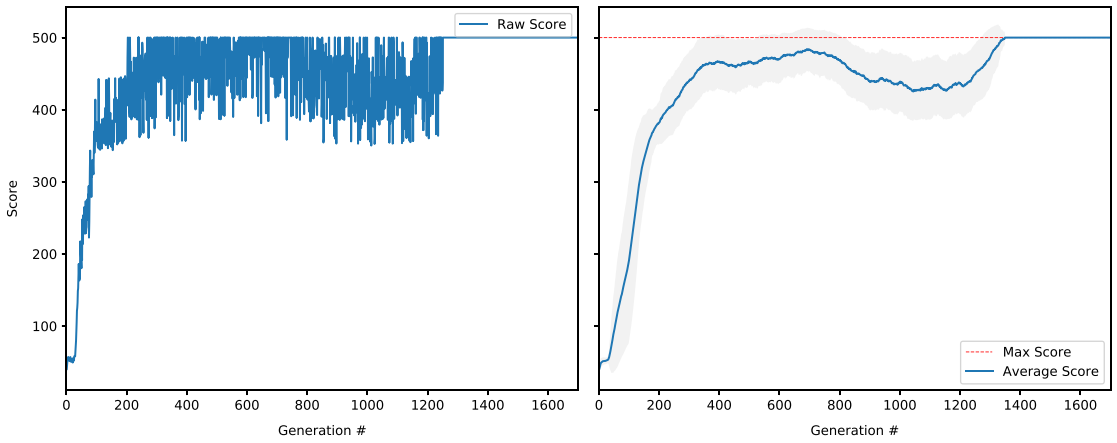

Figure 8. Performance of the VQC in the Cart-Pole environment. Left panel: average score of the top-5 agents at each generation. Maximum steps allowed in this environment is 500. Right panel: gray area represents the standard deviation of rewards in each iteration during exploration with the standard RL reproducible setting for stability. Mean and standard deviation values are calculated from the raw scores (rewards) in the past 100 generations.

quantum circuit processing. This opens the possibility of studying other complex RL problems with quantum circuits via applying this dimension reduction technique. We set the number of generations to be 500, the population size $N = 500$, the truncation selection number $T = 10$, the mutation power $\sigma = 0.02$, the number of repetitions (for evaluating all the agents) $R_1 = 3$ and the number of repetitions (for evaluating the parents) $R_2 = 5$. For comparison, we provide the following three model configurations: TN-VQC with bond dimension $\chi = 1$, TN-VQC with bond dimension $\chi = 2$ and a baseline NN-VQC. The neural network in the baseline NN-VQC model is a single-layer fully connected NN, which is similar to the structure of the TN used in TN-VQC models. The performance of our hybrid TN-VQC model in the three MiniGrid environments, `MiniGrid-Empty-5x5-v0`, `MiniGrid-Empty-6x6-v0` and `MiniGrid-Empty-8x8-v0` are shown in figure 9.

In the `MiniGrid-Empty-5x5-v0` environment (with a maximum score of 0.955), the simplest of the three, it is clear that the average score of the top-5 agents is able to reach near-optimal value in less than 40 generations. It can be observed that the TN-VQC model with $\chi = 2$ converges faster and achieves superior performance than the other two models. In addition, both TN-VQC models demonstrate superior performance than the baseline NN-VQC model. In the `MiniGrid-Empty-6x6-v0` environment (with a maximum score of $\sim$0.956), which is harder than the previous one, we observe that for the best-performing model (TN-VQC with $\chi = 2$), the average score of the top-5 agents rises above 0.9 in 40 generations and reaches near-optimal value after around 120 generations. It can be observed that the NN-VQC model performance is inferior to the TN-VQC with $\chi = 2$. The TN-VQC with $\chi = 1$ produces the worst performance in this environment. In the `MiniGrid-Empty-8x8-v0` environment (with a maximum score of $\sim$0.961), the most difficult one among the three, it can be seen that for the TN-VQC model with $\chi = 2$, it takes about 350 generations for the average score of the top-5 agents to rise and stay steadily above 0.9. It is clear that our model performs the worst in the last environment in terms of both the convergence speed and the final scores. Surprisingly, the TN-VQC model with $\chi = 1$ performs best among the three models. It requires fewer generations to achieve a higher score.

9. Discussion

9.1. Relevant studies

Early work on quantum RL can be traced back to [111], which needs to load the environment into quantum superposition states. This is not generally applicable to *classical* environments. In [112], the authors consider the situation where computational agents are coupled to environments that are quantum-mechanical. More recent studies introduce and facilitate the use of VQCs in the applications of RL [29–33]. In contrast to the present study, they all use gradient-based methods to optimize the policy and/or value function. For the Cart-Pole testing environment, which has been studied in both [31, 32], we observed that these two works and ours all reach the optimal solution. However, our model requires only two qubits, fewer than the models in [31, 32], where a four-qubit VQC architecture is necessary. An interesting research direction is to what extent gradient-free or gradient-based actually affects the performance of VQC in RL problems. Could certain VQC architectures benefit more from a particular kind of optimization method? We will leave this for

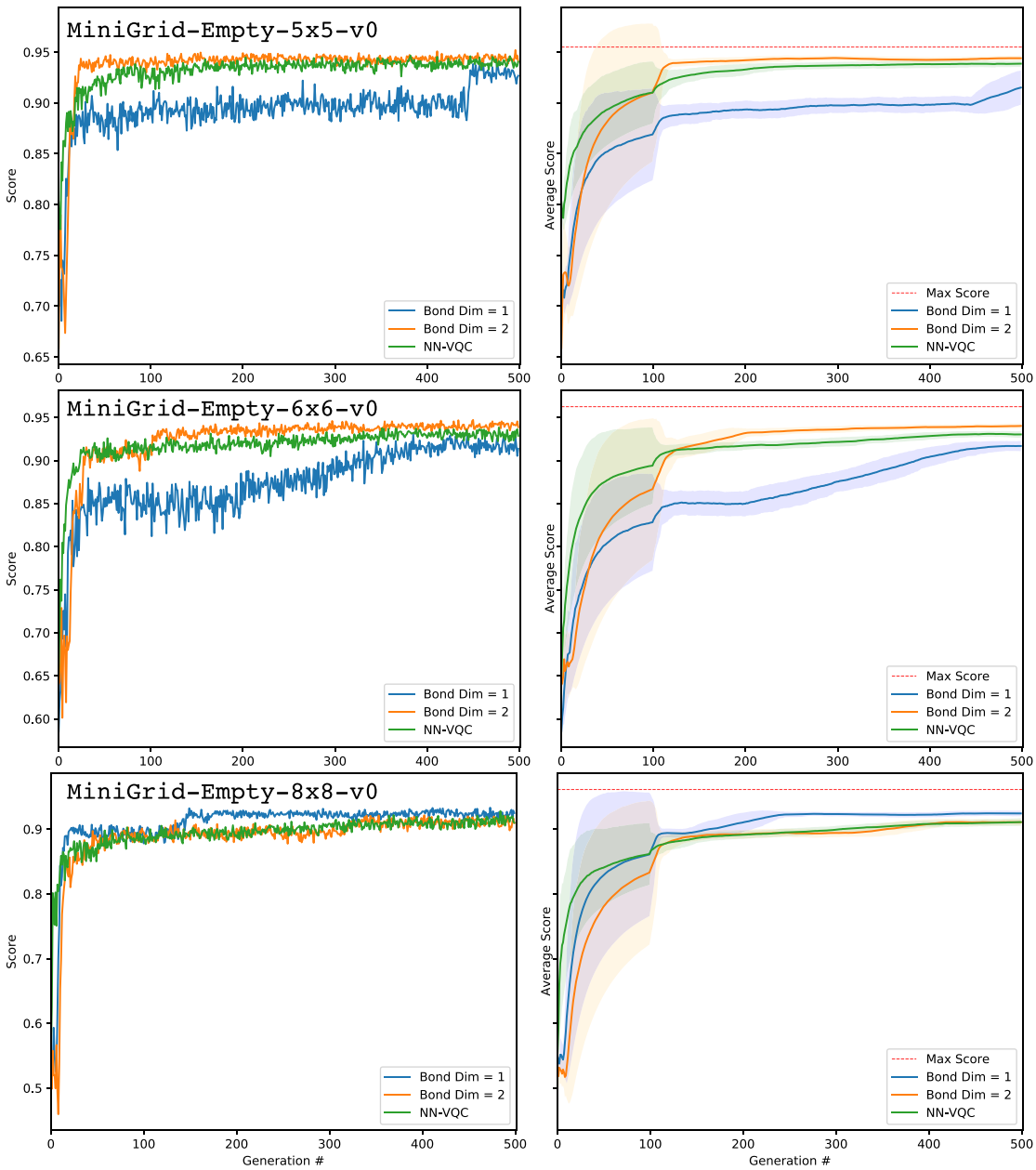

Figure 9. Performance of the hybrid TN-VQC in the MiniGrid environments. Left panels: average score of the top-5 agents at each generation. Right panels: gray area represents the standard deviation of rewards in each iteration during exploration with the standard RL reproducible setting for stability. Mean and standard deviation values are calculated from the raw scores (rewards) in the past 100 generations. We present three model configurations for each environment: TN-VQC with bond dim $\chi = 1$, TN-VQC with bond dim $\chi = 2$ and NN-VQC.

future investigation. For a more detailed review of recent developments in quantum RL, we refer interested readers to [69, 113].

9.2. Complex benchmarks

In this work, we further extend the complexity of the testing environments in comparison to previous works, including [29], which considers discrete observation/states. In particular, we largely push the boundaries of quantum RL via incorporating quantum-inspired TN into a VQC-based architecture. Despite its success, there is still a significant gap between current quantum RL and classical RL in terms of the capability of processing high-dimensional inputs. Future research directions may include the application of various TN architectures, such as MERA, PEPS or tree tensor networks for processing complex or high-dimensional inputs. For example, a recent development of the MERA-based quantum convolutional neural network [114] may inspire the next-generation design of quantum RL agents.

9.3. Evolutionary quantum circuits

The use of evolutionary methods in optimizing quantum circuits can also be found in these recent works [39, 115, 116]. Both [39, 115] use evolutionary methods to optimize VQE problems. Notably, [39] introduces an evolutionary approach involving structural mutation to optimize the quantum circuit. On the other hand, [116] utilizes a graph-encoding method to encode a quantum circuit and then adopts an evolutionary method to optimize the quantum model for certain classification tasks. However, none of these works considers the direct application of evolutionary optimization to quantum RL problems. Our work not only demonstrates the first successful implementation of an evolutionary method for quantum RL, but touches on another aspect rarely studied; the end-to-end training of a hybrid model consisting of MPS and VQC. Furthermore, the proposed framework is highly adjustable. For example, various TN architectures can be used to replace the MPS used in this paper to fulfill the need for different classical data, as well as changing QML tasks, such as classification, function approximation or sequential learning.

9.4. Techniques from classical neuroevolution

In the present study, we employ evolutionary algorithms specifically for optimizing the quantum circuit parameters. In classical neuroevolution, the whole architecture as well as the neural network parameters can be optimized through evolution. A framework called *NeuroEvolution of Augmenting Topologies* [117] has been proposed for evolving the classical neural network's topologies along with its weights. It is thus intriguing to investigate the prospect of applying these concepts to evolving QML architectures. In order to evolve a model with a complex architecture and a sizable number of parameters, it is crucial to encode the model itself in an efficient fashion. Recent advances in neuroevolution [118] could serve as guidance for designing high-performance evolutionary algorithms in QML.

A major issue yet to be addressed in our work is that our hybrid model can only achieve sub-optimal results on harder problems. In particular, it is challenging for the current model to achieve the maximum score when the rewards are sparse. One of the potential solutions to this issue is the *novelty search* developed in classical neuroevolution [119, 120]. The idea behind novelty search is that the agent is not trained to achieve the *objective* in a conventional way. Rather, novelty search rewards the agents that behave differently [121]. In classical deep RL problems, novelty search has been shown to be capable of solving hard RL problems with sparse rewards [122]. A potential direction for future research is thus to investigate whether these frameworks work in the quantum regime.

Evolutionary algorithms also play a critical role in the security of RL models. For example, in [123], the authors explore the potential of applying evolutionary algorithms to attacking a deep RL model. Hence, another potential research direction is to study the robustness of quantum RL agents [124].

9.5. Training on a real QC

Given the fact that currently available quantum devices are still very noisy, in this research we only consider the case of noise-free simulation. Although previous results [40–42] have indicated that VQC-based algorithms may be resilient to noise through the absorption of this undesirable effect into tunable parameters, with the limitation of current cloud-based quantum computing resources, it is impractical to implement the whole training process on a real quantum device to verify customized models such as the proposed TN-VQC. Furthermore, different noise mitigation schemes for TN-VQC are necessary and require further study. We expect that these issues can be resolved when commercial quantum devices become more reliable and accessible.

10. Conclusion

In this study, we present two quantum RL frameworks based on an evolutionary algorithm, of which one is purely quantum and the other has a hybrid quantum-classical architecture. In particular, we study two input loading schemes that can reduce the required qubit number of the VQC; amplitude encoding and TN compression, and demonstrate through numerical simulation the performance of each. First, through the Cart-Pole problem (with an input dimension of 4), we show that with amplitude encoding, a framework based on a VQC can provide a quantum advantage in terms of parameter savings. With a proper design of the quantum circuit, it is possible to reduce the number of parameters to the scale of $\text{poly}(\log(n))$, where n is the input dimension. Second, through the more complex MiniGrid problem (with an input dimension of 147), and with the incorporation of a TN-based dimensional reduction method, we show the possibility of compressing an input of a larger dimension into a representation that can be readily encoded into currently available quantum devices. Notably, this hybrid TN-VQC model can be trained in an end-to-end fashion, i.e. the TN and VQC parts are trained as a whole. The results of this work strongly suggest the prospect of these

versatile and scalable frameworks, which could shed light on the future design of RL algorithms for near-term QCs.

Data availability statement

The data that support the findings of this study are available upon reasonable request from the authors.

Acknowledgments

This work is supported by the U.S. Department of Energy, Office of Science, Office of High Energy Physics program under Award No. DE-SC-0012704 and the Brookhaven National Laboratory LDRD #20-024 (S Y-C C), the Ministry of Science and Technology (MOST) of Taiwan under Grant Nos. 107-2112-M-002-016-MY3, 108-2112-M-002-020-MY3 (Y-J K), 109-2112-M-002-023-MY3, 109-2627-M-002-003, 107-2627-E-002-001-MY3 and 109-2622-8-002-003 (H S G), the U.S. Air Force Office of Scientific Research under Award No. FA2386-20-1-4033, and by the National Taiwan University under Grant No. NTU-CC-110L890102 (H S G).

Appendix A. Quantum encoding

In order for the VQC to process classical data, it is necessary to encode the data into a quantum state. There are several kinds of encoding schemes commonly used in QML applications [86]. Different encoding methods provide varying extent of quantum advantage. Some of them are not readily implemented on real quantum hardware due to the large circuit depth. Here, we give the details of the two encoding methods used in this work.

A.1. Variational encoding

We describe the quantum operation behind the *variational encoding* used in the MiniGrid experiment. The initial quantum state $|0\rangle \otimes \cdots \otimes |0\rangle$ first undergoes the $H \otimes \cdots \otimes H$ operation to become an unbiased state $|+\rangle \otimes \cdots \otimes |+\rangle$. Consider an N-qubit system, the corresponding unbiased state is,

$$\begin{aligned}(H|0\rangle)^{\otimes N} &= \underbrace{H|0\rangle \otimes \cdots \otimes H|0\rangle}_{N} \\ &= \underbrace{|+\rangle \otimes \cdots \otimes |+\rangle}_{N} \\ &= \underbrace{\left[\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right] \otimes \cdots \otimes \left[\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right]}_{N} \\ &= \frac{1}{\sqrt{2^N}}(|0\rangle + |1\rangle)^{\otimes N} \\ &= \sum_{(q_1,q_2,\ldots,q_N)\in\{0,1\}^N} \frac{1}{\sqrt{2^N}}|q_1\rangle \otimes |q_2\rangle \otimes \cdots \otimes |q_N\rangle. \end{aligned} \tag{A1}$$

This unbiased quantum state subsequently goes through the *encoding* part, which consists of R_y and R_z rotations. These rotation operations are parameterized by the compressed representation vector $\mathbf{x} = (x_1, x_2, \ldots, x_8)$. For the ith qubit we choose the R_y and R_z rotation angles to be $\arctan(x_i)$ and $\arctan(x_i^2)$, respectively, where $i = 1, 2, \ldots, 8$. After this encoding process, the next step is to go through the *variational* parts as described in the main text.

A.2. Amplitude encoding

We elucidate the quantum operation behind the *amplitude encoding* used in the Cart-Pole experiment. We adopt the method presented in [87], which is further demonstrated for QML applications in [86]. The general goal of amplitude encoding is to encode a vector $(\alpha_0, \ldots, \alpha_{2^n-1})$ into an n-qubit quantum state $|\Psi\rangle = \alpha_0|00\cdots0\rangle + \cdots + \alpha_{2^n-1}|11\cdots1\rangle$, where α_i are real numbers and the vector $(\alpha_0, \ldots, \alpha_{2^n-1})$ is normalized. This can be achieved by inversely running the quantum routine, which transforms an arbitrary quantum state $|\Psi\rangle = \alpha_0|00\cdots0\rangle + \cdots + \alpha_{2^n-1}|11\cdots1\rangle$ into the state $|00\cdots0\rangle$. In [87], the authors demonstrate the quantum routine to perform this transformation via a sequence of multi-controlled

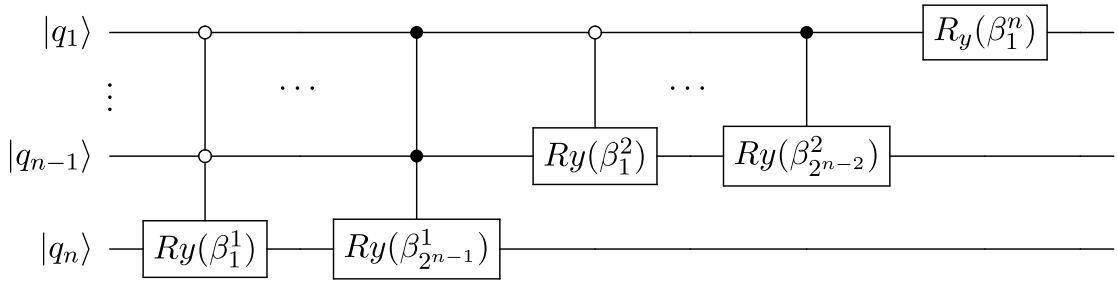

Figure 10. Generic quantum routine for amplitude encoding (before inversion).

rotations. Here, we follow the presentation in [86]. Consider an n-qubit system, the generic multi-controlled rotations around the vectors v^i with angles β_i on the last qubit q_n consists of the sequential operations of the following 2^{n-1} gates:

$$\begin{aligned} &c_{q_1=0}\cdots c_{q_{n-1}=0}R_{q_n}\left(v^1,\beta_1\right) \quad |q_1\ldots q_{n-1}\rangle\,|q_n\rangle\,,\\ &c_{q_1=0}\cdots c_{q_{n-1}=1}R_{q_n}\left(v^2,\beta_2\right) \quad |q_1\ldots q_{n-1}\rangle\,|q_n\rangle\,,\\ &\vdots\\ &c_{q_1=1}\cdots c_{q_{n-1}=1}R_{q_n}\left(v^{2^{n-1}},\beta_{2^{n-1}}\right) \quad |q_1\ldots q_{n-1}\rangle\,|q_n\rangle\,. \end{aligned} \tag{A2}$$

In the ML applications of interest, we consider only the case where all the amplitudes are real numbers. In this scenario, the quantum routine includes a *cascade* of multi-controlled R_y rotations [86]. The *cascade* of operations refers to sequentially implemented multi-controlled rotations on each qubit q_i in the system. For example, the operation on the last qubit q_n is shown in equation (A2). In an n-qubit system (see figure 10 for illustration), the first operation in the cascade is the one on qubit q_n,

$$\begin{aligned} &c_{q_1=0}\cdots c_{q_{n-1}=0}R_{q_n}\left(v^1,\beta_1^1\right) \quad |q_1\ldots q_{n-1}\rangle\,|q_n\rangle\,,\\ &c_{q_1=0}\cdots c_{q_{n-1}=1}R_{q_n}\left(v^2,\beta_2^1\right) \quad |q_1\ldots q_{n-1}\rangle\,|q_n\rangle\,,\\ &\vdots\\ &c_{q_1=1}\cdots c_{q_{n-1}=1}R_{q_n}\left(v^{2^{n-1}},\beta_{2^{n-1}}^1\right) \quad |q_1\ldots q_{n-1}\rangle\,|q_n\rangle\,, \end{aligned} \tag{A3}$$

where the rotation axes $v^1,\ldots,v^{2^{n-1}}$ are y-axis (R_y rotation). The second operation in the cascade is implemented on the qubit q_{n-1} in a similar way,

$$\begin{aligned} &c_{q_1=0}\cdots c_{q_{n-2}=0}R_{q_{n-1}}\left(v^1,\beta_1^2\right) \quad |q_1\ldots q_{n-2}\rangle\,|q_{n-1}\rangle\,,\\ &c_{q_1=0}\cdots c_{q_{n-2}=1}R_{q_{n-1}}\left(v^2,\beta_2^2\right) \quad |q_1\ldots q_{n-2}\rangle\,|q_{n-1}\rangle\,,\\ &\vdots\\ &c_{q_1=1}\cdots c_{q_{n-2}=1}R_{q_{n-1}}\left(v^{2^{n-2}},\beta_{2^{n-2}}^2\right) \quad |q_1\ldots q_{n-2}\rangle\,|q_{n-1}\rangle\,. \end{aligned} \tag{A4}$$

The rotation angles β_j^s can be shown to be [87],

$$\beta_j^s = 2\,\arcsin\left(\frac{\sqrt{\sum_{l=1}^{2^{s-1}}\left|\alpha_{(2j-1)2^{s-1}+l}\right|^2}}{\sqrt{\sum_{l=1}^{2^s}\left|\alpha_{(j-1)2^s+l}\right|^2}}\right), \tag{A5}$$

where α_i represents each of the amplitudes $(\alpha_0,\ldots,\alpha_{2^n-1})$. To utilize the aforementioned quantum routine to perform *amplitude encoding*, we can simply invert each and every operation and apply them in reverse order on the initial quantum state $|00\cdots0\rangle$ [86]. We provide the example for the two-qubit system used in our Cart-Pole experiment in figure 11.

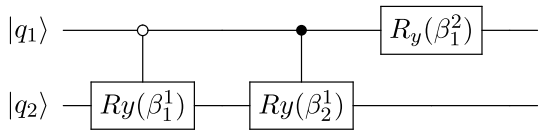

Figure 11. Quantum routine for amplitude encoding (before inversion) for the Cart-Pole experiment.

Appendix B. Quantum circuit evolution algorithm

Algorithm 1. Evolutionary quantum deep Q learning

Define the mutation power σ
Define the number of generation M
Define the number of truncation selection T
Define the repetition number of playing (all agents) R_1
Define the repetition number of playing (top agents)R_2
Initialize the population $\mathcal{P}$ with N individuals/agents
Initialize action-value function quantum circuit Q with random parameters $\boldsymbol{\theta}$ for each individual/agent
for generation $g = 1, 2, \ldots, M$ **do**
  for individual/agent $i = 1, 2, \ldots, N$ **do**
    for $r = 1, 2, \ldots, R_1$ **do**
      Reset the testing environment
      Reset the cumulative reward (score) $S_{i,r} \leftarrow 0$
      for $t = 1, 2, \ldots, K$ **do**
        The agent selects $a_t = \max_a Q^*(s_t, a; \boldsymbol{\theta})$ from the output of the quantum circuit
        Execute action a_t in emulator and observe reward r_t and next state s_{t+1}
        Record the reward $S_{i,r} \leftarrow S_{i,r} + r_t$
      end for
    end for
    Output the average score of the agent after playing R_1 times as $S_i^{avg} = \frac{1}{R_1}\sum_{r=1}^{R_1} S_{i,r}$
  end for
  Output the top T agents according to their average scores S_i^{avg}
  for $c = 1, 2, \ldots, N-1$ **do**
    Randomly select an agent from the top T agents
    Mutate the parameter vector $\boldsymbol{\theta}$ according to the rule $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \sigma\epsilon$ where the mutation is Gaussian $\epsilon \sim \mathcal{N}(0, I)$
  end for
  for agent in top T agents $j = 1, 2, \ldots, T$ **do**
    Playing the game R_2 times
    Record the average score over playing R_2 times E_j^{avg}
  end for
  Keep the best agent according to the average scores E_j^{avg} as the Nth children which is the elite
end for

ORCID iDs

Samuel Yen-Chi Chen https://orcid.org/0000-0003-0114-4826
Hsi-Sheng Goan https://orcid.org/0000-0001-8117-5846
Ying-Jer Kao https://orcid.org/0000-0002-3329-6018

References

[1] Grzesiak N *et al* 2020 Efficient arbitrary simultaneously entangling gates on a trapped-ion quantum computer *Nat. Commun.* **11** 1–6
[2] Arute F *et al* 2019 Quantum supremacy using a programmable superconducting processor *Nature* **574** 505–10
[3] Cross A 2018 The IBM Q experience and QISkit open-source quantum computing software *APS Meeting Abstracts*
[4] Harrow A W and Montanaro A 2017 Quantum computational supremacy *Nature* **549** 203–9
[5] Nielsen M A and Chuang I 2002 *Quantum Computation and Quantum Information* (Cambridge: Cambridge University Press)
[6] Shor P W 1999 Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer *SIAM Rev.* **41** 303–32
[7] Grover L K 1997 Quantum mechanics helps in searching for a needle in a haystack *Phys. Rev. Lett.* **79** 325
[8] Preskill J 2018 Quantum computing in the NISQ era and beyond *Quantum* **2** 79
[9] Gottesman D 1997 Stabilizer codes and quantum error correction (arXiv:quant-ph/9705052)
[10] Gottesman D 1998 Theory of fault-tolerant quantum computation *Phys. Rev.* A **57** 127

[11] Cerezo M *et al* 2020 Variational quantum algorithms (arXiv:2012.09265)
[12] Bharti K *et al* 2021 Noisy intermediate-scale quantum (NISQ) algorithms (arXiv:2101.08448)
[13] Mitarai K, Negoro M, Kitagawa M and Fujii K 2018 Quantum circuit learning *Phys. Rev.* A **98** 032309
[14] Li W and Deng D-L 2021 Recent advances for quantum classifiers (arXiv:2108.13421)
[15] Sutton R S and Barto A G 2018 *Reinforcement Learning: An Introduction* (Cambridge, MA: MIT Press)
[16] Mnih V *et al* 2 2015 Human-level control through deep reinforcement learning *Nature* **518** 529–33
[17] Schrittwieser J *et al* 2019 Mastering Atari, Go, chess and shogi by planning with a learned model (arXiv:1911.08265)
[18] Badia A P, Piot B, Kapturowski S, Sprechmann P, Vitvitskyi A, Guo D and Blundell C 2020 Agent57: outperforming the Atari human benchmark (arXiv:2003.13350)
[19] Kapturowski S, Ostrovski G, Quan J, Munos R and Dabney W 2018 Recurrent experience replay in distributed reinforcement learning *Int. Conf. on Learning Representations*
[20] Silver D *et al* 2016 Mastering the game of Go with deep neural networks and tree search *Nature* **529** 484–9
[21] Silver D *et al* 2017 Mastering the game of Go without human knowledge *Nature* **550** 354–9
[22] Fösel T, Tighineanu P, Weiss T and Marquardt F 2018 Reinforcement learning with neural networks for quantum feedback *Phys. Rev.* X **8** 031084
[23] Kuo E-J, Fang Y-L L and Chen S Y-C 2021 Quantum architecture search via deep reinforcement learning (arXiv:2104.07715)
[24] Sivak V, Eickbusch A, Liu H, Royer B, Tsioutsios I and Devoret M 2021 Model-free quantum control with reinforcement learning (arXiv:2104.14539)
[25] Sweke R, Kesselring M S, van Nieuwenburg E P L and Eisert J 2018 Reinforcement learning decoders for fault-tolerant quantum computation (arXiv:1810.07207)
[26] Liu Y-H and Poulin D 2018 Neural belief-propagation decoders for quantum error-correcting codes (arXiv:1811.07835)
[27] Poulsen Nautrup H, Delfosse N, Dunjko V, Briegel H J and Friis N 2018 Optimizing quantum error correction codes with reinforcement learning (arXiv:1812.08451)
[28] Melnikov A A, Poulsen Nautrup H, Krenn M, Dunjko V, Tiersch M, Zeilinger A and Briegel H J 2018 Active learning machine learns to create new quantum experiments *Proc. Natl Acad. Sci.* **115** 1221–6
[29] Chen S Y-C, Yang C-H H, Qi J, Chen P-Y, Ma X and Goan H-S 2020 Variational quantum circuits for deep reinforcement learning *IEEE Access* **8** 141007–24
[30] Lockwood O and Si M 2020 Reinforcement learning with quantum variational circuit *Proc. AAAI Conf. on Artificial Intelligence and Interactive Digital Entertainment* vol 16 pp 245–51
[31] Jerbi S, Gyurik C, Marshall S, Briegel H J and Dunjko V 2021 Variational quantum policies for reinforcement learning (arXiv:2103.05577)
[32] Skolik A, Jerbi S and Dunjko V 2021 Quantum agents in the Gym: a variational quantum algorithm for deep Q-learning (arXiv:2103.15084)
[33] Wu S, Jin S, Wen D and Wang X 2020 Quantum reinforcement learning in continuous action space (arXiv:2012.10711)
[34] Such F P, Madhavan V, Conti E, Lehman J, Stanley K O and Clune J 2017 Deep neuroevolution: genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning (arXiv:1712.06567)
[35] Chevalier-Boisvert M, Willems L and Pal S 2018 Minimalistic gridworld environment for OpenAI Gym (available at: https://github.com/maximecb/gym-minigrid)
[36] Brockman G, Cheung V, Pettersson L, Schneider J, Schulman J, Tang J and Zaremba W 2016 OpenAI Gym (arXiv:1606.01540)
[37] Schuld M, Bergholm V, Gogolin C, Izaac J and Killoran N 2019 Evaluating analytic gradients on quantum hardware *Phys. Rev.* A **99** 032331
[38] Kyriienko O and Elfving V E 2021 Generalized quantum circuit differentiation rules (arXiv:2108.01218)
[39] Franken L, Georgiev B, Muecke S, Wolter M, Piatkowski N and Bauckhage C 2020 Gradient-free quantum optimization on NISQ devices (arXiv:2012.13453)
[40] Kandala A, Mezzacapo A, Temme K, Takita M, Brink M, Chow J M and Gambetta J M 2017 Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets *Nature* **549** 242–6
[41] Farhi E, Goldstone J and Gutmann S 2014 A quantum approximate optimization algorithm (arXiv:1411.4028)
[42] McClean J R, Romero J, Babbush R and Aspuru-Guzik A 2016 The theory of variational hybrid quantum-classical algorithms *New J. Phys.* **18** 023023
[43] Chen S Y-C, Yoo S and Fang Y-L L 2020 Quantum long short-term memory (arXiv:2009.01783)
[44] Paine A E, Elfving V E and Kyriienko O 2021 Quantum quantile mechanics: solving stochastic differential equations for generating time-series (arXiv:2108.03190)
[45] Kyriienko O, Paine A E and Elfving V E 2021 Solving nonlinear differential equations with differentiable quantum circuits *Phys. Rev.* A **103** 052416
[46] Schuld M, Bocharov A, Svore K and Wiebe N 2018 Circuit-centric quantum classifiers (arXiv:1804.00633)
[47] Havlíček V, Córcoles A D, Temme K, Harrow A W, Kandala A, Chow J M and Gambetta J M 2019 Supervised learning with quantum-enhanced feature spaces *Nature* **567** 209–12
[48] Farhi E and Neven H 2018 Classification with quantum neural networks on near term processors (arXiv:1802.06002)
[49] Benedetti M, Lloyd E, Sack S and Fiorentini M 2019 Parameterized quantum circuits as machine learning models *Quantum Sci. Technol.* **4** 043001
[50] Mari A, Bromley T R, Izaac J, Schuld M and Killoran N 2019 Transfer learning in hybrid classical-quantum neural networks (arXiv:1912.08278)
[51] Abohashima Z, Elhosen M, Houssein E H and Mohamed W M 2020 Classification with quantum machine learning: a survey (arXiv:2006.12270)
[52] Easom-McCaldin P, Bouridane A, Belatreche A and Jiang R 2020 Towards building a facial identification system using quantum machine learning techniques (arXiv:2008.12616)
[53] Sarma A, Chatterjee R, Gili K and Yu T 2019 Quantum unsupervised and supervised learning on superconducting processors (arXiv:1909.04226)
[54] Stein S A, Baheri B, Tischio R M, Chen Y, Mao Y, Guan Q, Li A and Fang B 2020 A hybrid system for learning classical data in quantum states (arXiv:2012.00256)
[55] Chen S Y-C, Huang C-M, Hsing C-W and Kao Y-J 2021 An end-to-end trainable hybrid classical-quantum classifier *Mach. Learn.: Sci. Technol.* **2** 045021

[56] Chen S Y-C, Wei T-C, Zhang C, Yu H and Yoo S 2020 Quantum convolutional neural networks for high energy physics data analysis (arXiv:2012.12177)
[57] Wu S L *et al* 2021 Application of quantum machine learning using the quantum variational classifier method to high energy physics analysis at the LHC on IBM quantum computer simulator and hardware with 10 qubits *J. Phys. G: Nucl. Part. Phys.* **48** 125003
[58] Stein S A, Mao Y, Baheri B, Guan Q, Li A, Chen D, Xu S and Ding C 2021 QuClassi: a hybrid deep neural network architecture based on quantum state fidelity (arXiv:2103.11307)
[59] Chen S Y-C, Wei T-C, Zhang C, Yu H and Yoo S 2021 Hybrid quantum-classical graph convolutional network (arXiv:2101.06189)
[60] Jaderberg B, Anderson L W, Xie W, Albanie S, Kiffner M and Jaksch D 2021 Quantum self-supervised learning (arXiv:2103.14653)
[61] Mattern D, Martyniuk D, Willems H, Bergmann F and Paschke A 2021 Variational quanvolutional neural networks with enhanced image encoding (arXiv:2106.07327)
[62] Hur T, Kim L and Park D K 2021 Quantum convolutional neural network for classical data classification (arXiv:2108.00661)
[63] Qi J, Yang C-H H and Chen P-Y 2021 QTN-VQC: an end-to-end learning framework for quantum neural networks (arXiv:2110.03861)
[64] Dallaire-Demers P-L and Killoran N 2018 Quantum generative adversarial networks *Phys. Rev.* A **98** 012324
[65] Stein S A, Baheri B, Tischio R M, Mao Y, Guan Q, Li A, Fang B and Xu S 2020 QuGAN: a generative adversarial network through quantum states (arXiv:2010.09036)
[66] Zoufal C, Lucchi A and Woerner S 2019 Quantum generative adversarial networks for learning and loading random distributions *npj Quantum Inf.* **5** 1–9
[67] Situ H, He Z, Li L and Zheng S 2018 Quantum generative adversarial network for generating discrete data (arXiv:1807.01235)
[68] Nakaji K and Yamamoto N 2020 Quantum semi-supervised generative adversarial network for enhanced data classification (arXiv:2010.13727)
[69] Jerbi S, Trenkwalder L M, Nautrup H P, Briegel H J and Dunjko V 2021 Quantum enhancements for deep reinforcement learning in large spaces *PRX Quantum* **2** 010328
[70] Chen C-C, Shiba K, Sogabe M, Sakamoto K and Sogabe T 2020 Hybrid quantum-classical Ulam-von Neumann linear solver-based quantum dynamic programing algorithm *Proc. Annual Conf. of JSAI* vol JSAI2020 p 2K6ES203
[71] Kwak Y, Yun W J, Jung S, Kim J-K and Kim J 2021 Introduction to quantum reinforcement learning: theory and pennylane-based implementation (arXiv:2108.06849)
[72] Nagy D, Tabi Z, Hága P, Kallus Z and Zimborás Z 2021 Photonic quantum policy learning in OpenAI Gym (arXiv:2108.12926)
[73] Bausch J 2020 Recurrent quantum neural networks (arXiv:2006.14619)
[74] Takaki Y, Mitarai K, Negoro M, Fujii K and Kitagawa M 2020 Learning temporal data with variational quantum recurrent neural network (arXiv:2012.11242)
[75] Abbaszade M, Salari V, Mousavi S S, Zomorodi M and Zhou X 2021 Application of quantum natural language processing for language translation *IEEE Access* **9** 130434–48
[76] Di Sipio R, Huang J-H, Chen S Y-C, Mangini S and Worring M 2021 The dawn of quantum natural language processing (arXiv:2110.06510)
[77] Yang C-H H, Qi J, Chen S Y-C, Chen P-Y, Siniscalchi S M, Ma X and Lee C-H 2021 Decentralizing feature extraction with quantum convolutional neural network for automatic speech recognition *ICASSP 2021—2021 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE) pp 6523–7
[78] Lloyd S, Schuld M, Ijaz A, Izaac J and Killoran N 2020 Quantum embeddings for machine learning (arXiv:2001.03622)
[79] Nghiem N A, Chen S Y-C and Wei T-C 2021 Unified framework for quantum classification *Phys. Rev. Res.* **3** 033056
[80] Qi J and Tejedor J 2021 Classical-to-quantum transfer learning for spoken command recognition based on quantum neural networks (arXiv:2110.08689)
[81] Chen S Y-C and Yoo S 2021 Federated quantum machine learning *Entropy* **23** 460
[82] Chehimi M and Saad W 2021 Quantum federated learning with quantum data (arXiv:2106.00005)
[83] Sim S, Johnson P D and Aspuru-Guzik A 2019 Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms *Adv. Quantum Technol.* **2** 1900070
[84] Lanting T *et al* 2014 Entanglement in a quantum annealing processor *Phys. Rev.* X **4** 021041
[85] Du Y, Hsieh M-H, Liu T and Tao D 2018 The expressive power of parameterized quantum circuits (arXiv:1810.11922)
[86] Schuld M and Petruccione F 2018 Information encoding *Supervised Learning with Quantum Computers* (Cham: Springer International Publishing) pp 139–71
[87] Möttönen M, Vartiainen J J, Bergholm V and Salomaa M M 2005 Transformation of quantum states using uniformly controlled rotations *Quantum Inf. Comput.* **5** 467–73
[88] White S R 1992 Density matrix formulation for quantum renormalization groups *Phys. Rev. Lett.* **69** 2863
[89] White S R 1993 Density-matrix algorithms for quantum renormalization groups *Phys. Rev.* B **48** 10345
[90] Eisert J 2013 Entanglement and tensor network states (arXiv:1308.3318)
[91] Cirac J I and Verstraete F 2009 Renormalization and tensor product states in spin chains and lattices *J. Phys. A: Math. Theor.* **42** 504004
[92] Verstraete F, Murg V and Cirac J I 2008 Matrix product states, projected entangled pair states and variational renormalization group methods for quantum spin systems *Adv. Phys.* **57** 143–224
[93] Schollwöck U 2005 The density-matrix renormalization group *Rev. Mod. Phys.* **77** 259
[94] Schollwöck U 2011 The density-matrix renormalization group in the age of matrix product states *Ann. Phys., NY* **326** 96–192
[95] Perez-Garcia D, Verstraete F, Wolf M M and Cirac J I 2007 Matrix product state representations (arXiv:quant-ph/0608197)
[96] Verstraete F, Porras D and Cirac J I 2004 Density matrix renormalization group and periodic boundary conditions: a quantum information perspective *Phys. Rev. Lett.* **93** 227205
[97] Biamonte J and Bergholm V 2017 Tensor networks in a nutshell (arXiv:1708.00006)
[98] Stoudenmire E and Schwab D J 2016 Supervised learning with tensor networks *Advances in Neural Information Processing Systems* vol 29 pp 4799–807
[99] Liu D, Ran S-J, Wittek P, Peng C, García R B, Su G and Lewenstein M 2019 Machine learning by unitary tensor network of hierarchical tree structure *New J. Phys.* **21** 073059
[100] Stoudenmire E M 2018 Learning relevant features of data with multi-scale tensor networks *Quantum Sci. Technol.* **3** 034003
[101] Glasser I, Pancotti N and Cirac J I 2020 From probabilistic graphical models to generalized tensor networks for supervised learning *IEEE Access* **8** 68169–82

[102] Efthymiou S, Hidary J and Leichenauer S 2019 Tensornetwork for machine learning (arXiv:1906.06329)
[103] Glasser I, Pancotti N and Cirac J I 2018 Supervised learning with generalized tensor networks (arXiv:1806.05964)
[104] Bhatia A S, Saggi M K, Kumar A and Jain S 2019 Matrix product state–based quantum classifier *Neural Comput.* **31** 1499–517
[105] Han Z-Y, Wang J, Fan H, Wang L and Zhang P 2018 Unsupervised generative modeling using matrix product states *Phys. Rev.* X **8** 031012
[106] Cheng S, Wang L, Xiang T and Zhang P 2019 Tree tensor networks for generative modeling *Phys. Rev.* B **99** 155131
[107] Sun Z-Z, Peng C, Liu D, Ran S-J and Su G 2020 Generative tensor network classification model for supervised machine learning *Phys. Rev.* B **101** 075135
[108] Bradley T-D, Stoudenmire E M and Terilla J 2020 Modeling sequences with quantum states: a look under the hood *Mach. Learn.: Sci. Technol.* **1** 035008
[109] Huggins W, Patil P, Mitchell B, Whaley K B and Stoudenmire E M 2019 Towards quantum machine learning with tensor networks *Quantum Sci. Technol.* **4** 024001
[110] Ran S-J 2020 Encoding of matrix product states into quantum circuits of one-and two-qubit gates *Phys. Rev.* A **101** 032310
[111] Dong D, Chen C, Li H and Tarn T-J 2008 Quantum reinforcement learning *IEEE Trans. Syst. Man Cybern.* B **38** 1207–20
[112] Dunjko V, Taylor J M and Briegel H J 2015 Framework for learning agents in quantum environments (arXiv:1507.08482)
[113] Dunjko V, Taylor J M and Briegel H J 2017 Advances in quantum reinforcement learning *2017 IEEE Int. Conf. on Systems, Man and Cybernetics (SMC)* (IEEE) pp 282–7
[114] Cong I, Choi S and Lukin M D 2019 Quantum convolutional neural networks *Nat. Phys.* **15** 1273–8
[115] Anand A, Degroote M and Aspuru-Guzik A 2020 Natural evolutionary strategies for variational quantum computation (arXiv:2012.00101)
[116] Lu Z, Shen P-X and Deng D-L 2020 Markovian quantum neuroevolution for machine learning (arXiv:2012.15131)
[117] Stanley K O and Miikkulainen R 2002 Efficient evolution of neural network topologies *Proc. 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No. 02TH8600)* vol 2 (IEEE) pp 1757–62
[118] Zhang H, Yang C-H H, Zenil H, Kiani N A, Shen Y and Tegner J N 2020 Evolving neural networks through a reverse encoding tree *2020 IEEE Congress on Evolutionary Computation (CEC)* (IEEE) pp 1–10
[119] Lehman J and Stanley K O 2010 Efficiently evolving programs through the search for novelty *Proc. 12th Annual Conf. on Genetic and Evolutionary Computation* pp 837–44
[120] Risi S, Hughes C E and Stanley K O 2010 Evolving plastic neural networks with novelty search *Adapt. Behav.* **18** 470–91
[121] Lehman J and Stanley K O 2011 Abandoning objectives: evolution through the search for novelty alone *Evol. Comput.* **19** 189–223
[122] Conti E, Madhavan V, Such F P, Lehman J, Stanley K O and Clune J 2017 Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents (arXiv:1712.06560)
[123] Yang C-H H, Qi J, Chen P-Y, Ouyang Y, Hung I-T D, Lee C-H and Ma X 2020 Enhanced adversarial strategically-timed attacks against deep reinforcement learning *ICASSP 2020—2020 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE) pp 3407–11
[124] Yang C-H H, Hung I, Danny T, Ouyang Y and Chen P-Y 2021 Causal inference Q-network: toward resilient reinforcement learning (arXiv:2102.09677)