Continuous Variational Quantum Algorithms for Time Series

1st Muhao Guo
School of ECEE
Arizona State University
Tempe, USA
mguo26@asu.edu

2nd Yang Weng*
School of ECEE
Arizona State University
Tempe, USA
yang.weng@asu.edu

3rd Lili Ye
School of ECEE
Arizona State University
Tempe, USA
liliye@asu.edu

4th Ying Cheng Lai
School of ECEE
Arizona State University
Tempe, USA
Ying-Cheng.Lai@asu.edu

2023 International Joint Conference on Neural Networks (IJCNN) | 978-1-6654-8867-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/IJCNN54540.2023.10191609

***Abstract*—Variational quantum algorithms (VQAs) are the leading algorithm for achieving quantum advantage using near-term quantum computers. VQAs use parameterized quantum circuits for inference, and the variational parameters in quantum circuits can be trained using a classical optimizer. The parameters are trained to guide how the quantum bits evolve and make the final measurements closely match the ground truth. However, this way of learning from raw data makes it difficult to capture the underlying dynamic information in the data, especially for time series data. To address this limitation, we proposed continuous variational quantum algorithms (CVQAs) for time series in this paper. CVQAs use quantum variational circuits to parameterize the dynamics of time series, thus they can learn the dynamic information behind the data. After the dynamics are trained, the prediction results will be obtained by a differential equation solver working on the dynamics. Since we aim to model the dynamics of data instead of the data itself, the quantum circuit in our approach will need fewer qubits and variational gates. To evaluate our proposed approach, we compare our model with baseline models on several weather time series. Experimental results prove that our approach has better or equivalent results but with fewer qubits and variational gates compared to baseline models.**

***Index Terms*—Quantum machine learning, Variational quantum circuit, Dynamics, Time series**

I. Introduction

Quantum machine learning (QML) has been attracting attention recently because they are able to achieve a significantly better effective dimension than comparable classical ones [1], [2]. QMLs utilize the laws of quantum computing (QC), such as superposition and entanglement, to perform high-dimensional computations. This leads to quantum advantages in terms of better simulations, faster computations, or network performance.

However, existing available quantum hardware implements only a few hundred physical bits. Even worse, the information carried by the quantum bits is easy to be swamped by noise if the gates have long sequences. Due to these limitations, many of the expected algorithms, such as Shor's [4], remain out of reach. As a result, people focus their attention on noisy intermediate-scale quantum (NISQ) devices. There is a growing belief that NISQ devices may find useful applications and commercialization in the near future [5], [6]. As prototypes of quantum computers are made available to researchers for experimentation, algorithmic research is adapting to match the pace of hardware development [3].

To be executed on NISQ devices, some effective QML algorithms that are suitable for small-scale quantum systems have been proposed [7]. Currently, the simplest and most efficient approach to benefit from near-term quantum computers is to use variational quantum circuits (VQCs) [8]–[10]. This kind of algorithm is called a variational quantum algorithm (VQA), which belongs to the hybrid quantum-classical approach. The quantum component is implemented by a variational quantum circuit whose output depends on a tunable parameterized gate in the circuit. The classical component, on the other hand, utilizes the classical optimization tools to optimize the VQAs by tuning the parameters. Thus, the variational quantum circuits run on a quantum computer, while the optimization function is performed on a classical computer.

VQA is particularly suitable for implementing QMLs and has an immediate impact on many applications. For instance, [39] and [12] showed VQAs can be successful in classification tasks. [13] enhances a conventional NN architecture by adding a variational quantum layer that outperforms its classical equivalent. [14] reshapes classical deep reinforcement learning algorithms like experience replay and target network into a representation of variational quantum circuits. [22] proposed the hybrid quantum-classical model of LSTM and showed this quantum version of LSTM converges faster, or equivalently, reaches a better accuracy than its classical counterpart. [23] designed a hybrid quantum-classical recurrent neural network that aims at solving time series prediction problems.

Recently, many methods have been studied in the field of machine learning for time series prediction [15]–[17]. A particularly challenging field is represented by the weather forecast because small errors grow into larger errors, causing the model to diverge from the actual weather over time. It becomes more difficult to predict when rapid instability occurs [18]. Deterministic systems struggle to interpret stochastic fluctuations hence quantum computing may provide a more robust approach to address the problem [23].

Quantum computing has had many applications in time series prediction [19]–[23], but at present, especially regarding weather time series, there are still few works that demonstrate

the practical advantages of classical quantum hybrid networks. Moreover, the extant quantum-classical hybrid models emphasize the concept of layers and fit the observations of the data one by one, which makes it difficult to learn the overall continuous dynamic information implicitly behind the data points.

In this work, we introduce a novel continuous variational quantum algorithm (CVQA) for time series prediction. CVQAs aim to extract and learn the dynamics behind the time series. Unlike current VQAs, which try to fit a fixed set of discrete points, CVQAs learn the underlying continuous dynamics of the data, which can make the model more powerful and easier to generalize to unknown data.

The core idea of CVQAs is to use a variational quantum circuit to parameterize a vector field of the time series. The variational quantum circuit takes the current state of the system as input and produces the time derivative of that state as output. The parameterized gate in the VQC determines the quantum state evolution over time. The output will be obtained by quantum measurement at the end of the circuit, which reflects the final state of the time series evolution. The entire procedure represents how the time series system will change over time. Integrating the vector field over time makes it possible to calculate the system's trajectories and make predictions about its future behavior. Thus, our approach learns the continuous trajectories of the system evolution. Since the VQC is used to parameterize the dynamics, the size of the VQC is smaller than directly applying it to time series data. The structure of our approach is shown in Fig 2.

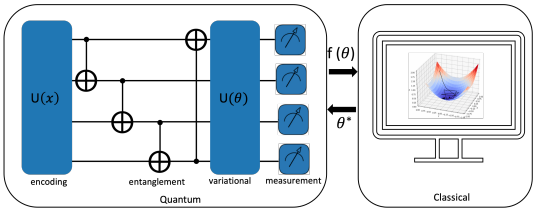

Fig. 1. Variational quantum algorithm (VQA)

II. Related Work

A. Variational Quantum Circuits (VQCs)

The machine learning tasks can be improved when carried out on a quantum computer since quantum information processing is expected to bring us exponential speedups on some problems [24]. However, due to the limitations of current quantum computer hardware, many algorithms can only stay in the theoretical stage, such as [4]. Therefore, avoiding high-depth quantum circuits is one of the key factors to achieving quantum advantage at this stage. In this context, hybrid classical-quantum algorithms consisting of quantum circuits of relatively low depth have been proposed. For example, the quantum variational solver (QVE) [25], [26] and the quantum approximate optimization algorithm [27]–[29] (QAOA).

Variational quantum circuits [8]–[10] are a pervasive class of hybrid algorithms. It allows one to train quantum machine learning algorithms in the same way one trains neural networks. By adjusting the parameter θ iteratively, the expected value relative to the ansatz state $|\psi(\theta)\rangle$ will be optimized. A gradient-based systematic optimization of parameters is introduced for the tuning parameters in the quantum circuit, just like the backpropagation method utilized in feed-forward neural networks [8].

B. Continuous Deep Learning Architectures

Continuous deep learning architectures have recently reemerged as Neural Ordinary Differential Equations (Neural ODEs) and their variants [30]–[34]. This infinitely deep approach theoretically bridges the gap between deep learning and dynamical systems, providing a new perspective [35]. According to [30], the scalar-valued loss with respect to all inputs of any ODE solver can be computed directly without backpropagating through the operations of the ODE solver. The intermediate quantities of the forward pass will not need to be stored. It means the Neural ODEs can be trained with a constant memory cost. Because of the limited quantum sources, memory efficiency is more valuable in the quantum field. However, it is still an unknown field regarding how to design a continuous QML model.

Meanwhile, the way of working by solving differential equations to obtain feasible solutions makes continuous deep learning suitable for learning dynamics, particularly for time series. Quantum computing works on the high dimensions and provides a new perspective for time series forecasting [36]–[38], which inspires us to design a quantum version of a continuous model for time series.

III. Preliminary

A. Variational Quantum Algorithms (VQAs)

VQAs are a kind of hybrid quantum-classical algorithm that effectively executes QML algorithms on NISQ devices. Designing a quantum circuit has a certain resemblance to designing a neural network. For example, we can use VQC to fit a function, similar to what a traditional neural network is designed to do [45]. For a VQC, during the feed-forward process, VQCs use a parameterized quantum circuit to prepare quantum states and use measurements on final states to obtain observable results. The quantum state is described by a wave function, which has a probability amplitude that represents the probability of each possible state. The wave function is used to make predictions about the behavior of a quantum system, and its properties are determined by the Schrödinger equation [51]. During the backpropagation process, classical optimization is used to find the best set of parameters that minimize the cost function.

The parameterized quantum circuit always includes the encoding layer, entanglement layer, variational layer, and measurement layer. The encoding layer corresponds to the input layer to encode the classical data in the quantum circuit. The entanglement layer creates quantum entanglement among qubits, which is typically realized by the CNOT gates. The encoding and entanglement layers are important because

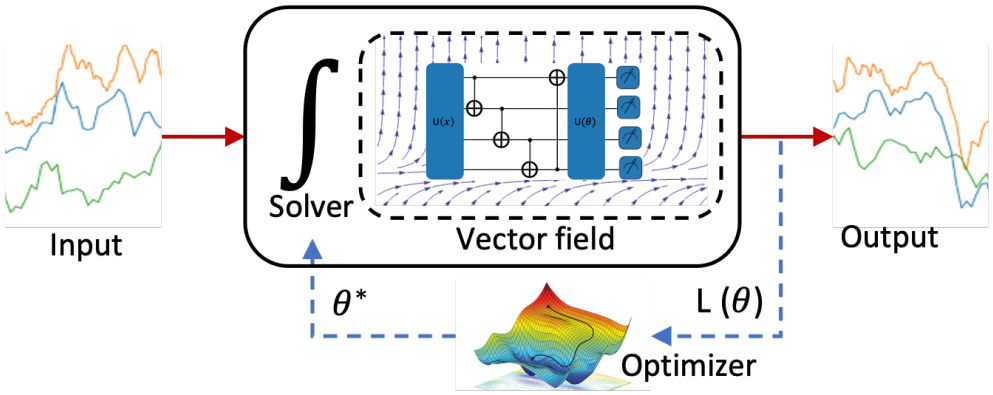

Fig. 2. The structure of CVQAs. In the center of CVQAs, the vector field is parameterized by a VQC to learn the dynamics of time series. The solver integrates the vector field over time to obtain the final state of the time series.

generally, they can create a single superposition quantum state and multi-particle entangled states, respectively, which realize the exponential speed up and parallel operation on the qubits. The variational layer includes parametrized quantum gates to optimize the output, which is similar to a neural network but with different functional forms. The entanglement and variational layers can be regarded as one standard layer and can be repeated to achieve the learning goal, sometimes they are regarded just as one variational layer [22]. In the end, the measurement layer extracts classical information from the quantum output.

There are several ways to encode the classical data into the quantum states [46]–[48]. A direct way is to convert a single integer into a binary number and make each binary bit corresponds to a qubit. The number of qubits is $O(Nlog(M))$, where N is the number of input elements and M is the input integer. However, this approach requires many quantum bits when the input elements and M become large, which may destroy the quantum advantage. To avoid this situation, we use the rotation gate for encoding. A rotation gate $R_i(\phi)$ is a single qubit rotation through an angle ϕ around the i axis on the Bloch sphere, where $i \in \{x, y, z\}$. In this way, the number of qubits needed is $O(N)$, thus it is the so-called scaled encoding [45].

In the variational layer, the parameters are typically real numbers that are encoded into the quantum circuit using gates that depend on them. These gates are often called "parameterized gates". Similar to the rotation gate used in the encoding layer, $R_x(\theta)$, $R_y(\theta)$, and $R_z(\theta)$ are three typically parameterized gates, which represent the single qubit rotation through an angle θ around the x-axis, y-axis, and z-axis, respectively. The difference between the rotation gate used in the quantum encoding layer and the quantum variational layer is that in the encoding layer, the rotation gate is used to map classical data into a quantum state, whereas, in the variational layer, the rotation gate is used to perform a specific computation and its parameters can be optimized to improve the performance of the circuit. These parametrized gates change the starting wave function and determine the final quantum state, which is used as a preliminary solution to the problem. The subsequent processing of the quantum state leads to the final solution to the given problem.

At the end of the VQCs, one can extract information from a quantum state through the measurement layer. Measurement in quantum computing is to obtain the outcome from a quantum state by collapsing the quantum state into one of its eigenstates. When a quantum system is measured, the wave function collapses into one of its eigenstates, and the corresponding eigenvalue provides information about the state of the quantum system [49], [50]. Measurements in quantum computing are typically performed by applying a Hermitian operator, which represents an observable property of the quantum system. Pauli operators such as Pauli X, Pauli Y, and Pauli Z gates are typically used as measurement operators. Because the computational basis of Pauli Z is common [45], we use the Pauli Z gate.

We illustrate the working process of a variational quantum circuit. Suppose x is the input classical data, $|0\rangle$ is the initial quantum state, and $U(x)$ is the quantum gate we use to encode the classical data in the encoding layer. In the beginning, x is encoded into a quantum state $|\psi_{in}(x)\rangle$ through using a unitary gate U, as shown in Equation. 1.

$$|\psi_{in}(x)\rangle = U(x)\,|0\rangle\,. \tag{1}$$

After the encoding layer, the variational layer is applied to the quantum state $|\psi_{in}(x)\rangle$. Specifically, another unitary gate $U(\theta)$ is applied into $|\psi_{in}(x)\rangle$, where the θ is the parameter that can be tuned. An output state $|\psi_{out}(x,\theta)\rangle$ will be generated after the variational layer:

$$|\psi_{out}(x,\theta)\rangle = U(\theta)\,|\psi_{in}(x)\rangle\,. \tag{2}$$

At the end of the circuit, one can measure the expectation values of some chosen observables. For example, we can use a subset of the Pauli operators $\{B\} \subset \{I, X, Y, Z\}^{\otimes N}$ for the measurement, where N represents the number of qubits of the

circuit. The expectation value of an observable $\hat{O}$ in the state $|\psi_{out}(x,\theta)\rangle$ is defined as:

$$\langle O_{\theta,x}\rangle = \langle\psi_{out}(x,\theta)|\,\hat{O}\,|\psi_{out}(x,\theta)\rangle \tag{3}$$

$$= \langle\psi_{in}(x)|\,U^{\dagger}(\theta)\hat{O}U(\theta)\,|\psi_{in}(x)\rangle\,. \tag{4}$$

By using a classical output function l, such as a linear map, one can obtain the final output $\hat{y}$:

$$\hat{y}_{\theta,x} = l(\langle O\rangle). \tag{5}$$

After the execution of the VQC, a cost function $C(\hat{y}_{\theta,x}, y)$ is evaluated, and the parameter θ is updated with the help of an optimizer in a classical computer. Specifically, these parameters are adjusted by minimizing the cost C. The VQCs can be trained using gradient-based algorithms, as other neural networks. The way to get the partial derivatives by evaluating parameter-shifted instances of a variational circuit is called the parameter-shift rule [8], [39]:

$$\frac{\partial\langle O_{\theta,x}\rangle}{\partial\theta} = \frac{1}{2}(\langle O_{\theta+\frac{\pi}{2},x}\rangle - \langle O_{\theta-\frac{\pi}{2},x}\rangle). \tag{6}$$

Thus, gradients can be backpropagated in the quantum layer, and the entire hybrid quantum-classical network can be trained using the backpropagation method.

B. Continuous Deep Learning Model for Dynamics

Neural ODEs are recently popular continuous deep learning models. They can learn the dynamics of a continuous system by solving a set of ordinary differential equations (ODEs). This makes them particularly powerful for modeling systems that evolve over time, such as time series data or physical systems. One of the key advantages of Neural ODEs is that they can learn complex, non-linear dynamics using a continuous-time formulation, which allows them to capture intricate patterns in the data that might be missed by traditional neural networks. Additionally, because they are based on ODEs, they can be easily integrated over time, which makes them well-suited for tasks that require the prediction of future states or behaviors. Neural ODEs can be interpreted as a continuous version of Residual Networks [40]. Recall the formulation of a residual network:

$$h_{t+1} - h_t = f(h_t, \theta_f), \tag{7}$$

where the f is the residual block and the θ_f represents the parameters of f. The left side of Equation 7 can be seen as a denominator of 1, so it can be represented by $\frac{h_{t+1}-h_t}{1} = f(h_t,\theta_f)$. When the number of layers becomes infinitely large, and the step becomes infinitely small, Equation 7 will become an ordinary differential equation format, as shown in Equation 8.

$$\lim_{dt\to 0}\frac{h_{t+dt}-h_t}{dt} = \frac{dh(t)}{dt} = f(h(t),t,\theta_f). \tag{8}$$

Thus, the NODE will have the same format as an ODE:

$$h'(t) = f(h(t),t,\theta_f), \quad h(0) = x_0, \tag{9}$$

where x_0 is the input data. Typically, function f will be some standard simple neural architecture, such as a Multilayer perception (MLP). The θ_f represents trainable parameters in f.

To obtain any final state of $h(t)$ when $t = T$, all that is needed is to solve an ordinary differential equation with initial values, which is called an initial value problem (IVP):

$$h(T) = h(0) + \int_0^T f(h(t),t,\theta_f)dt. \tag{10}$$

Thus, a Neural ODE can transform from $h(0)$ to $h(T)$ through the solutions to the IVP of the ODE.

By the properties of ODEs, Neural ODEs are always invertible; we can reverse the limits of integration, or alternatively, integrate $-f$. The *Adjoint Sensitivity Method* [41] based on reverse-time integration of an expanded ODE, allows for finding gradients of the initial value problem solutions $h(T)$ with respect to parameters θ_f and the initial values $h(0)$ [30]. The forward evaluation and backpropagation process of a Neural ODE model can be calculated by using a black-box differential equation solver, called ODE solvers [53]–[55].

ODE solvers can be divided into fixed step size solvers and adaptive step size solvers, and both of them can be used for Neural ODEs. Given a final time T, a fixed step size solver will choose the time t_i from $[0,T]$ where $\Delta t = t_{i+1} - t_i$. Δt is fixed in advance and independent of i. An adaptive step solver, such as *Runge-Kutta Method* [55] is a relatively modern solver, which can vary the size of the next step so that the error made during the solver is approximately equal to some tolerance.

IV. Methods

A. Continuous Variational Quantum Algorithms (CVQAs)

The basic idea of CVQAs is to use a variational quantum circuit to parameterize the vector fields in a dynamic system and use the ODE solver to solve the potential ODEs of the system.

Suppose the input data is x, and the variational quantum circuit used for parameterizing the vector fields is denoted as U_v. After the input data passes U_v, we can obtain the quantum state

$$|\psi_{out}(x,\theta,t)\rangle = U_v(\theta,x,t)\left|0^{\otimes N_v}\right\rangle, \tag{11}$$

where the N_v is the number of qubits used in U_v.

At the end of U_v, we measure the expectation values of a chosen observable $\hat{A}$. Thus, we obtain the measurement values in terms of time t:

$$\langle A\rangle_{\theta,x,t} = \left\langle 0^{\otimes N_v}\right| U_v^{\dagger}(\theta,x,t)\hat{A}U_v(\theta,x,t)\left|0^{\otimes N_v}\right\rangle. \tag{12}$$

To reshape the output so that it has the same dimension as the input data, we use a linear map l and obtain the output. We denote the output as g' since it is the output of a vector field:

$$g'(\theta,x,t) = l\circ\langle A\rangle_{\theta,x,t}. \tag{13}$$

We use θ_l to denote the parameters in l. We see this problem as an IVP of the ODE. The final state of $g(t)$ is the solution.

Algorithm 1 Training CVQAs

Input: Training Dataset $\mathcal{D}(x, y)$
Initialize QVC $U_v(\theta)$, θ_l, iteration
for iteration **do**
  Eq. (11): execute the QVC $U_v(\theta)$ on data x
  $\langle A\rangle_{\theta,x,t} \leftarrow$ measures the QVC using an observation
  $l \circ \langle A\rangle_{\theta,x,t} \leftarrow$ apply linear map l to reshape the readout of measurement
  g(T) $\leftarrow$ ODESolver(x, Eq. (13), θ, θ_l)
  calculate the MSE loss: C = MSE(g(T), y)
  θ, θ_l $\leftarrow$ Adam(C, Parameter-Shift-Rule($\langle A\rangle_{\theta,x,t}$), l) update θ, θ_l
end for
return θ, θ_l

When $t = T$, we obtain the final state of the ODE as the solution:

$$g(T) = x_{t_0} + \int_0^T g'(\theta, x, t)dt \tag{14}$$

$$= x_{t_0} + \int_0^T l \circ \langle A\rangle_{\theta,x,t} dt, \tag{15}$$

where x_{t_0} represents the input data x at $t = 0$, as the initial state of the ODE.

B. Training CVQAs

We want to use a backward propagation algorithm to optimize all the parameters, as the general VQAs. A key factor is how to find the gradient. On one hand, the *Adjoint Sensitivity Method* [41] allows for finding gradients of the initial value problem solutions $g(T)$ with respect to parameters θ_g and the initial values x [30].

On the other hand, the parameter-shift rule [8], [39] works by adding a small shift to the parameters of the quantum circuit and measuring the corresponding change in the expectation value of the observable. This allows for the gradient of the expectation value to be estimated, which can then be used in gradient-based optimization algorithms to update the parameters of the quantum circuit. All these techniques offer the possibility of using gradient-based backward propagation algorithms to optimize our model. We combine these techniques and train our model by using the classical optimizer Adam [56]. The training process of CVQAs is shown in Algorithm 1.

V. Experiment and Analysis

A. Environment Setup

All the models were implemented in Python 3.9 and realized in PyTorch. The machine used for the experiments was provided with an NVIDIA GeForce GTX 1070. All quantum models are based on PennyLane [44] with a simple state simulator of qubit-based quantum circuit architecture. We trained each dataset for 10 epochs and used a batch size of 16. All the models are trained with the Adam algorithm, using a learning rate of 10^{-2}. Mean squared error (MSE) is used as the loss function. To visually compare the prediction results, we designed a prediction accuracy evaluation criterion as follows:

$$Accuracy = 1 - \sqrt{\frac{1}{N}\sum_{i=1}^{N} E_i^2}, \tag{16}$$

where $E_i = \frac{|y_i - \hat{y_i}|}{y_i}$ is the relative error. To avoid possible experimental errors, we ran each experiment three times for each model and recorded the results according to mean and standard values.

B. Model and Baselines Setup

CVQAs model uses a variational quantum circuit U_v for parameterizing the vector field of the dynamic of a time series. To verify the effectiveness of our approach, we modeled the vector field with two different ansatzes and evaluated their performance, separately. As shown in Fig. 3, (a) is one qubit ansatz, it uses the angle encoding layer realized by X rotation gate $R_x(x)$. The variational layer is realized by another X rotation gate $R_x(\theta)$. Thus, (a) has one adjustable parameter and two one-qubit gates. (b) is four qubits-ansatz. It also uses the angle encoding layer, which is realized by X rotation gates $R_x(x)$. We use the CNOT gate for the purpose of entanglement. The variational layer is also performed by X rotation gates $R_x(\theta)$. (b) has four adjustable parameters, eight one-qubit gates, and four two-qubit gates.

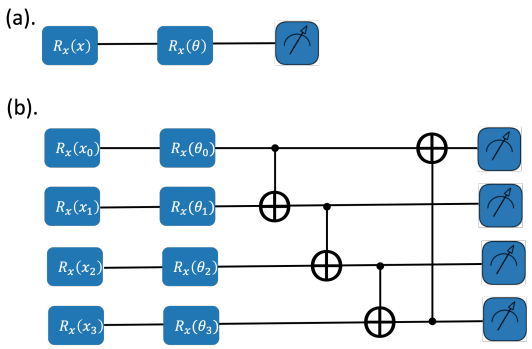

Fig. 3. Ansatz of vector fields in CVQAs. (a) uses one qubit to parameterize the vector field, which includes one rotation gate in the encoding layer and one rotation gate in the variational layer. (b) uses four qubits to parameterize the vector field, which includes four rotation gates in the encoding layer, four CNOT gates in the entanglement layer, and four rotation gates in the variational layer.

The baseline models we used are Quantum Neural Networks (QNNs) realized by the general variational quantum algorithms, and the quantum version LSTM (QLSTM) [22]. For a fair comparison, we design two QNNs, which have the exact same structures of the ansatzes as (a) and (b), respectively.

For the QLSTM, four quantum circuits are used as four weights calculations, as a simplified version of [22]. For each quantum circuit in the QLSTM, we use the same ansatz as shown in Fig 3 (b). It is worth noting that since QLSTM is implemented by four quantum circuits, the size of QLSTM is four times larger than that of our model of (b). The QLSTM

uses a total of $4 \times 4 = 16$ qubits, $4 \times 4 = 16$ rotation gates for encoding, $4 \times 4 = 16$ CNOT gates for entanglement, and $4 \times 4 = 16$ rotation gates for the variational layers. A QLSTM has 16 adjustable parameters in the ansatz.

C. Dataset

To evaluate our model, we compare different models' performances on the daily climate time series data [42], which contains weather data collected from the city of Delhi over the period of four years from 2013 to 2017. This data is collected from Weather Underground API. We evaluate four weather indicators including mean temperature, humidity, wind speed, and mean pressure. The mean temperature was averaged from multiple 3-hour intervals in a day. The units of humidity value for the day are grams of water vapor per cubic meter volume of air. Wind speed is measured in kilometers per hour. The pressure is measured in atmospheric pressure. For all experiments, the training set is the data from January 1, 2013 to December 31, 2016. The test set is the data from January 1, 2017 to April 24, 2017. We designed a window to be the length of 4, i.e., to predict the current value based on four days of historical data.

D. Prediction Results and Analysis

We first implemented our model with the ansatz of (a). We compared the QNN with the same ansatz of (a) and the QLSTM with our model. The mean and standard values of the training loss were recorded for three runs. As shown in Fig. 4, CVQAs converged with a relatively low loss compared to the QNN and QLSTM during the training. This is more evident in the top left and bottom right graphs. The top left, top right, bottom left, and bottom right represent the mean temperature, humidity, wind speed, and mean pressure, respectively.

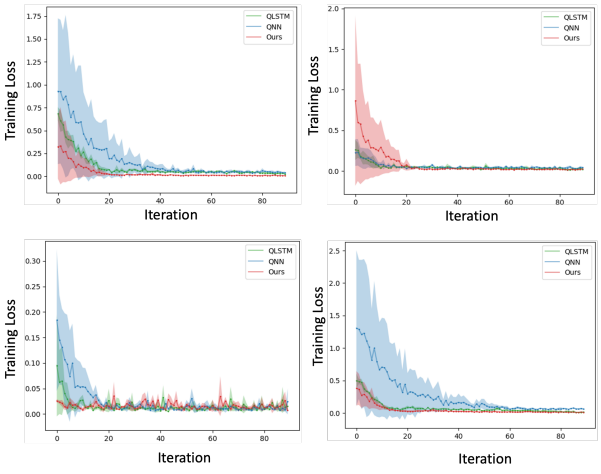

Fig. 4. The loss of different models during training. CVQAs have a low training loss compared to the baseline models, especially in the mean temperature and mean pressure datasets.

We recorded the training loss and the accuracy based on Equation 16 for the four datasets, as shown in Table I, Table II, Table III, and Table IV, respectively.

TABLE I
EXPERIMENT RESULT OF MEAN TEMPERATURE

	Training Loss	Accuracy
QNN	0.026 ± 0.008	0.815 ± 0.125
QLSTM	0.018 ± 0.004	0.917 ± 0.002
CVQA(Ours)	0.004 ± 0.000	0.919 ± 0.002

TABLE II
EXPERIMENT RESULT OF HUMIDITY

	Training Loss	Accuracy
QNN	0.011 ± 0.001	0.820 ± 0.018
QLSTM	0.009 ± 0.004	0.829 ± 0.019
CVQA(Ours)	0.007 ± 0.001	0.829 ± 0.014

Taking the mean temperature and humidity as examples, we tested our model on time series. After training ten epochs, CVQAs show a strong prediction ability as shown in Fig. 5.

We then extend the model to 4 qubits, as shown in Fig. 3(b), and compare it to a QNN that is also 4 qubits. We also recorded the mean value and standard values over three runs. As shown in Fig. 6, for the same number of qubits, our model has a smaller loss compared to QNN.

E. Size of CVQAs

In the era of NISQ, VQAs are always subject to the model size. VQAs with more qubits and deeper quantum circuits will suffer from more series noise effects. Thus, designing VQAs with a small model size is valuable. In this section, we compare different models in terms of the number of qubits, the number of single-qubit gates, and the number of two-qubit gates, i.e., the CNOT gates, and the number of variable parameters in the ansatz.

As shown in Fig.7, CVQAs are smaller in size compared to QLSTM and QNN but can achieve similar or better results. An intuitive reason is that modeling the dynamic behavior of time series is more effortless than modeling the time series itself. CVQAs aim to use a quantum circuit to model the dynamics

TABLE III
EXPERIMENT RESULT OF WIND SPEED

	Training Loss	Accuracy
QNN	0.004 ± 0.001	-1.479 ± 0.138
QLSTM	0.003 ± 0.000	-0.630 ± 0.039
CVQA(Ours)	0.004 ± 0.001	-0.684 ± 0.323

TABLE IV
EXPERIMENT RESULT OF MEAN PRESSURE

	Training Loss	Accuracy
QNN	0.039 ± 0.005	0.995 ± 0.001
QLSTM	0.009 ± 0.007	0.998 ± 0.000
CVQA(Ours)	0.006 ± 0.002	0.998 ± 0.000

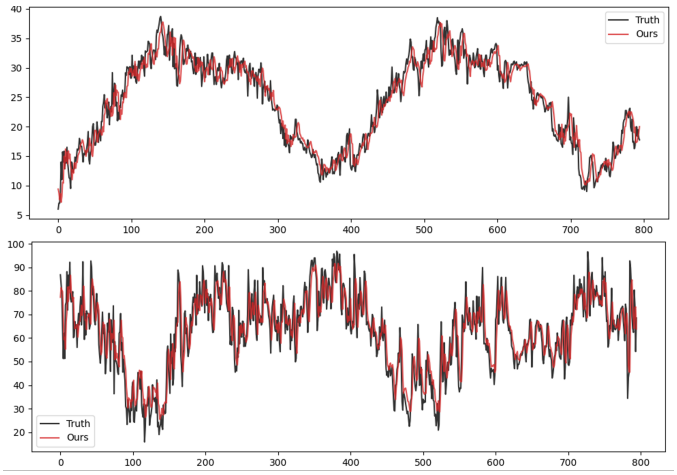

Fig. 5. CVQAs' predict results of mean temperature and humidity after training 10 epochs.

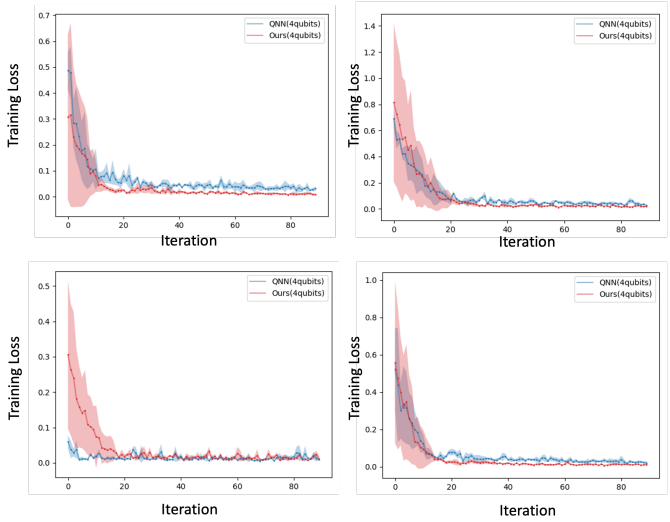

Fig. 6. The training loss of a CVQA with four qubits and a QNN with four qubits.

behind the time series, thus small-size quantum circuits are sufficient to achieve relatively high performance.

VI. Conclusion

The NISQ devices always limit quantum machine learning models, but using VQAs is the most straightforward way to realize the quantum advantages today. In this paper, we proposed a novel continuous VQA for dynamic learning in time series, named CVQAs. We use a quantum circuit to parameterize the vector field behind the dynamic system. Instead of learning the given input data directly, CVQAs aim to learn the dynamics behind the data utilizing a variational quantum circuit. Thus, CVQAs learn the implicit dynamic pattern from data better than other VQAs, particularly for weather time series data. Experiments show that our model can achieve equivalent or better accuracy but with fewer qubits and quantum gates than baseline models. With a small quantum

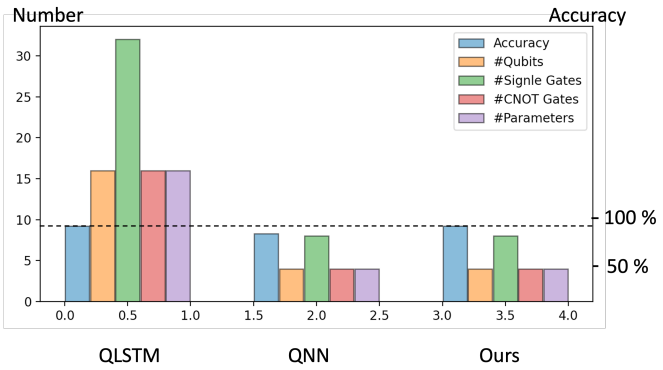

Fig. 7. A comprehensive comparison: the accuracy, the number of qubits, the number of variational quantum gates, and the number of CNOT gates of different models.

circuit size, our model will have more potential and value in the NISQ era.

References

[1] Abbas, A., Sutter, D., Zoufal, C., Lucchi, A., Figalli, A. and Woerner, S., 2021. The power of quantum neural networks. Nature Computational Science, 1(6), pp.403-409

[2] Wiebe, N., Kapoor, A. and Svore, K.M., 2014. Quantum deep learning. arXiv preprint arXiv:1412.3489.

[3] Benedetti, M., Lloyd, E., Sack, S. and Fiorentini, M., 2019. Erratum: parameterized quantum circuits as machine learning models (2019 Quant. Sci. Tech. 4 043001). Quantu m Science and Technology, 5(1), p.019601.

[4] Shor, P.W., 1994, November. Algorithms for quantum computation: discrete logarithms and factoring. In Proceedings 35th annual symposium on foundations of computer science (pp. 124-134). IEEE.

[5] Preskill, J., 2018. Quantum computing in the NISQ era and beyond. Quantum, 2, p.79.

[6] Mohseni, M., Read, P., Neven, H., Boixo, S., Denchev, V., Babbush, R., Fowler, A., Smelyanskiy, V. and Martinis, J., 2017. Commercialize quantum technologies in five years. Nature, 543(7644), pp.171-174.

[7] Bharti, K., Cervera-Lierta, A., Kyaw, T.H., Haug, T., Alperin-Lea, S., Anand, A., Degroote, M., Heimonen, H., Kottmann, J.S., Menke, T. and Mok, W.K., 2021. Noisy intermediate-scale quantum (NISQ) algorithms. arXiv preprint arXiv:2101.08448.

[8] Mitarai, K., Negoro, M., Kitagawa, M. and Fujii, K., 2018. Quantum circuit learning. Physical Review A, 98(3), p.032309.

[9] Tacchino, F., Barkoutsos, P.K., Macchiavello, C., Gerace, D., Tavernelli, I. and Bajoni, D., 2020, October. Variational learning for quantum artificial neural networks. In 2020 IEEE international conference on quantum computing and engineering (QCE) (pp. 130-136). IEEE.

[10] Cerezo, M., Arrasmith, A., Babbush, R., Benjamin, S.C., Endo, S., Fujii, K., McClean, J.R., Mitarai, K., Yuan, X., Cincio, L. and Coles, P.J., 2021. Variational quantum algorithms. Nature Reviews Physics, 3(9), pp.625-644.

[11] Schuld, M., Bocharov, A., Svore, K.M. and Wiebe, N., 2020. Circuit-centric quantum classifiers. Physical Review A, 101(3), p.032308. x

[12] 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(7747), pp.209-212.

[13] Sebastianelli, A., Zaidenberg, D.A., Spiller, D., Le Saux, B. and Ullo, S.L., 2021. On circuit-based hybrid quantum neural networks for remote sensing imagery classification. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 15, pp.565-580.

[14] 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, pp.141007-141024.

[15] Karevan, Z. and Suykens, J.A., 2020. Transductive LSTM for time-series prediction: An application to weather forecasting. Neural Networks, 125, pp.1-9.

[16] Hewage, P., Behera, A., Trovati, M., Pereira, E., Ghahremani, M., Palmieri, F. and Liu, Y., 2020. Temporal convolutional neural (TCN) network for an effective weather forecasting using time-series data from the local weather station. Soft Computing, 24(21), pp.16453-16482.

[17] Nielsen, A., 2019. Practical time series analysis: Prediction with statistics and machine learning. O'Reilly Media.

[18] Jain, H. and Jain, R., 2017, March. Big data in weather forecasting: Applications and challenges. In 2017 International conference on big data analytics and computational intelligence (ICBDAC) (pp. 138-142). IEEE.

[19] Azevedo, C.R. and Ferreira, T.A., 2007, August. Time series forecasting with qubit neural networks. In Conference: ASC'07 Proceedings of The Eleventh IASTED International.

[20] Ueguchi, T., Matsui, N. and Isokawa, T., 2016, September. Chaotic time series prediction by qubit neural network with complex-valued representation. In 2016 55th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE) (pp. 1353-1358). IEEE.

[21] Takaki, Y., Mitarai, K., Negoro, M., Fujii, K. and Kitagawa, M., 2021. Learning temporal data with a variational quantum recurrent neural network. Physical Review A, 103(5), p.052414.

[22] Chen, S.Y.C., Yoo, S. and Fang, Y.L.L., 2022, May. Quantum long short-term memory. In ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 8622-8626). IEEE.

[23] Ceschini, A., Rosato, A. and Panella, M., 2022, July. Hybrid Quantum-Classical Recurrent Neural Networks for Time Series Prediction. In 2022 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.

[24] Deffner, S. and Lutz, E., 2011. Nonequilibrium entropy production for open quantum systems. Physical review letters, 107(14), p.140404.

[25] Peruzzo, A., McClean, J., Shadbolt, P., Yung, M.H., Zhou, X.Q., Love, P.J., Aspuru-Guzik, A. and O'brien, J.L., 2014. A variational eigenvalue solver on a photonic quantum processor. Nature communications, 5(1), pp.1-7.

[26] 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(7671), pp.242-246.

[27] Farhi, E., Goldstone, J. and Gutmann, S., 2014. A quantum approximate optimization algorithm. arXiv preprint arXiv:1411.4028.

[28] Farhi, E. and Harrow, A.W., 2016. Quantum supremacy through the quantum approximate optimization algorithm. arXiv preprint arXiv:1602.07674.

[29] Otterbach, J.S., Manenti, R., Alidoust, N., Bestwick, A., Block, M., Bloom, B., Caldwell, S., Didier, N., Fried, E.S., Hong, S. and Karalekas, P., 2017. Unsupervised machine learning on a hybrid quantum computer. arXiv preprint arXiv:1712.05771.

[30] Chen, R.T., Rubanova, Y., Bettencourt, J. and Duvenaud, D.K., 2018. Neural ordinary differential equations. Advances in neural information processing systems, 31.

[31] Dupont, E., Doucet, A. and Teh, Y.W., 2019. Augmented neural odes. Advances in Neural Information Processing Systems, 32.

[32] Rubanova, Y., Chen, R.T. and Duvenaud, D.K., 2019. Latent ordinary differential equations for irregularly-sampled time series. Advances in neural information processing systems, 32.

[33] Kidger, P., Morrill, J., Foster, J. and Lyons, T., 2020. Neural controlled differential equations for irregular time series. Advances in Neural Information Processing Systems, 33, pp.6696-6707.

[34] Norcliffe, A., Bodnar, C., Day, B., Simidjievski, N. and Liò, P., 2020. On second order behaviour in augmented neural odes. Advances in Neural Information Processing Systems, 33, pp.5911-5921.

[35] Massaroli, S., Poli, M., Park, J., Yamashita, A. and Asama, H., 2020. Dissecting neural odes. Advances in Neural Information Processing Systems, 33, pp.3952-3963.

[36] Rivera-Ruiz, M.A., Mendez-Vazquez, A. and López-Romero, J.M., 2022. Time Series Forecasting with Quantum Machine Learning Architectures. In Mexican International Conference on Artificial Intelligence (pp. 66-82). Springer, Cham.

[37] Peters, E., Caldeira, J., Ho, A., Leichenauer, S., Mohseni, M., Neven, H., Spentzouris, P., Strain, D. and Perdue, G.N., 2021. Machine learning of high dimensional data on a noisy quantum processor. npj Quantum Information, 7(1), pp.1-5.

[38] Feng, Z.K., Niu, W.J., Tang, Z.Y., Jiang, Z.Q., Xu, Y., Liu, Y. and Zhang, H.R., 2020. Monthly runoff time series prediction by variational mode decomposition and support vector machine based on quantum-behaved particle swarm optimization. Journal of Hydrology, 583, p.124627.

[39] Schuld, M., Bergholm, V., Gogolin, C., Izaac, J. and Killoran, N., 2019. Evaluating analytic gradients on quantum hardware. Physical Review A, 99(3), p.032331.

[40] He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[41] Pontryagin, L.S., Boltyanskii, V.G., Gamkrelidze, R.V. and Mishchenko, E.F., 1961. Mathematical Theory of Optimal Processes [in Russian].

[42] Sumanthvrao. (2019). Daily Climate time series data, Version 1. Retrieved December 20, 2022 from https://www.kaggle.com/datasets/sumanthvrao/daily-climate-time-series-data?resource=download.

[43] Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. Neural computation, 9(8), pp.1735-1780.

[44] Bergholm, V., Izaac, J., Schuld, M., Gogolin, C., Alam, M.S., Ahmed, S., Arrazola, J.M., Blank, C., Delgado, A., Jahangiri, S. and McKiernan, K., 2018. Pennylane: Automatic differentiation of hybrid quantum-classical computations. arXiv preprint arXiv:1811.04968.

[45] Lockwood, O. and Si, M., 2020, October. Reinforcement learning with quantum variational circuit. In Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (Vol. 16, No. 1, pp. 245-251).

[46] Plesch, M. and Brukner, Č., 2011. Quantum-state preparation with universal gate decompositions. Physical Review A, 83(3), p.032302.

[47] Nakaji, K., Uno, S., Suzuki, Y., Raymond, R., Onodera, T., Tanaka, T., Tezuka, H., Mitsuda, N. and Yamamoto, N., 2022. Approximate amplitude encoding in shallow parameterized quantum circuits and its application to financial market indicators. Physical Review Research, 4(2), p.023136.

[48] Łodyga, J., Mazurek, P., Grudka, A. and Horodecki, M., 2015. Simple scheme for encoding and decoding a qubit in unknown state for various topological codes. Scientific reports, 5(1), p.8975.

[49] Wiseman, H.M. and Milburn, G.J., 2009. Quantum measurement and control. Cambridge university press.

[50] Aharonov, Y., Anandan, J. and Vaidman, L., 1993. Meaning of the wave function. Physical Review A, 47(6), p.4616.

[51] Berezin, F.A. and Shubin, M., 2012. The Schrödinger Equation (Vol. 66). Springer Science & Business Media.

[52] Kutta, W., 1901. Beitrag zur näherungsweisen Integration totaler Differentialgleichungen. Teubner.

[53] Wanner, G. and Hairer, E., 1996. Solving ordinary differential equations II (Vol. 375). New York: Springer Berlin Heidelberg.

[54] Kutta, W., 1901. Beitrag zur näherungsweisen Integration totaler Differentialgleichungen. Teubner.

[55] Runge, C., 1895. Über die numerische Auflösung von Differentialgleichungen. Mathematische Annalen, 46(2), pp.167-178.

[56] Kingma, D.P. and Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.