Journal of Physics Communications

OPEN ACCESS

RECEIVED
26 April 2024

REVISED
17 July 2024

ACCEPTED FOR PUBLICATION
9 August 2024

PUBLISHED
21 August 2024

PAPER

A variational approach to quantum gated recurrent units

Andrea Ceschini, Antonello Rosato and Massimo Panella*

Dept. of Information Engineering, Electronics and Telecommunications (DIET), University of Rome 'La Sapienza', Via Eudossiana 18, 00184 Rome, Italy

* Author to whom any correspondence should be addressed.

E-mail: andrea.ceschini@uniroma1.it, antonello.rosato@uniroma1.it and massimo.panella@uniroma1.it

Keywords: variational quantum circuits, quantum machine learning, hybrid quantum–classical neural network, deep learning, time series prediction

Abstract

Quantum Recurrent Neural Networks are receiving an increased attention thanks to their enhanced generalization capabilities in time series analysis. However, their performances were bottlenecked by long training times and unscalable architectures. In this paper, we propose a novel Quantum Recurrent Neural Network model based on Quantum Gated Recurrent Units. It uses a learnable Variational Quantum Layer to process temporal data, interspersed with two classical layers to properly match the dimensionality of the input and output vectors. Such an architecture has fewer quantum parameters than existing Quantum Long Short-Term Memory models. Both the quantum networks were evaluated on periodic and real-world time series datasets, together with the classical counterparts. The quantum models exhibited superior performances compared to the classical ones in all the test cases. The Quantum Gated Recurrent Units outperformed the Quantum Long Short-Term Memory network despite having a simpler internal configuration. Moreover, the Quantum Gated Recurrent Units network demonstrated to be about 25% faster during the training and inference procedure over the Quantum Long Short-Term Memory. This improvement in speed comes with one less quantum circuit to be executed, suggesting that our model may offer a more efficient alternative for implementing Quantum Recurrent Neural Networks on both simulated and real quantum hardware.

1. Introduction

In the past few years, Machine Learning (ML) and Deep Learning (DL) have imposed as primary tools to automate non-trivial tasks, achieving tremendous performances in tasks such as computer vision [1], natural language processing [2], sentiment analysis [3] and anomaly detection [4]. In this regard, effective time series prediction has rapidly emerged as a fundamental problem to be addressed and a variety of Neural Networks (NN) architectures have been proposed to this extent. Recurrent Neural Networks (RNN) are commonly employed for time series processing due to their chain-like structure and ability to model temporal dependencies in a given data sequence. Among them, Long Short-Term Memory (LSTM) [5] networks became a *de facto* standard, demonstrating remarkable performances in learning long-term dependencies without suffering from the vanishing gradient problem.

The major issue with LSTM networks is that they are computationally heavy due to their complex gate-based architecture [6]. For this reason, Gated Recurrent Unit (GRU) networks [7] were proposed as a variant of LSTM with comparable performances but simpler architecture and faster convergence [8]. GRU cells differ from LSTM because they lack the cell state c_t and are composed only of a reset gate r and an update gate z, as shown in figure 1. The reduced computational complexity of GRUs, which have one less gate than LSTMs, contributes to their faster performance. However, it is important to note that while this architectural difference can lead to more efficient computation, the maintenance of satisfactory results depends on various factors, including the specific application and data characteristics. Therefore, the faster computation enabled by the GRU's simpler structure

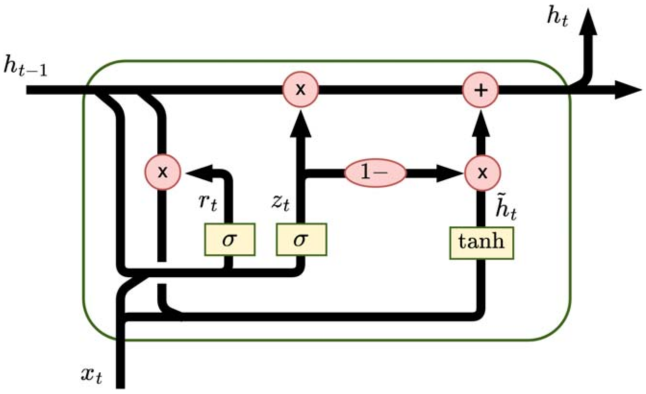

Figure 1. Internal structure of a classical GRU cell. Here, h_t represents the hidden state at the current time step t, and h_{t-1} represents the hidden state from the previous time step $t-1$. The candidate hidden state is represented by $\tilde{h}_t$, while x_t is the input vector at time t. The symbol σ denotes the sigmoid activation function, which is applied to the concatenated input vector $[h_{t-1}, x_t]$; the symbol tanh represents the hyperbolic tangent activation function; the 'x' mark indicates element-wise multiplication (i.e. Hadamard product); the '+' symbol denotes element-wise addition; the '1-' symbol represents the operation of subtracting the input vector from one, effectively inverting the values within the unit range.

does not inherently guarantee satisfactory results in all scenarios, but it has been empirically shown to perform well and even outperform LSTM in many cases [9, 10].

In recent times, Quantum Neural Networks (QNNs) have emerged as a rising trend in DL [11–13], with the expectation of superior performances against classical neural networks thanks to the leverage of quantum superposition and entanglement. QNNs are able to efficiently deal with large-scale heterogeneous data and perform effective high-dimensional processing with a reduced number of qubits, leading to faster computation and lower error rates [14, 15]. The technology behind QNNs goes from superconducting qubits [16] to photonic systems [17] and trapped-ion devices [18]. Since fault-tolerant quantum computers are still years away, researchers made significant advancement in the design of quantum algorithms meant to be executed on Noisy Intermediate-Scale Quantum (NISQ) devices.

Inspired by [19], in this work we propose a novel quantum variant of a GRU, which we denote as Quantum GRU (QGRU); to the best of our knowledge, it is the first implementation of a GRU in a variational quantum environment. The QGRU has 25% fewer quantum parameters with respect to the Quantum LSTM (QLSTM) and is about 25% faster both in the training and inference phase. In view of the current challenges associated with running quantum circuits on both simulators and real quantum devices, a less complex network architecture is more practical and feasible for the capabilities of contemporary quantum hardware. Moreover, we also perform a fair comparison between the proposed quantum networks and their classical counterparts, i.e. an LSTM and a GRU with the same magnitude of parameters. The forecasting ability of the networks is evaluated on three different datasets: a simple periodic function, a sunspots time series and the actual generation of wind energy in Italy for the month of June 2022. This is the first time that feasible quantum recurrent variational models are assessed on realistic problems. In fact, the second and third dataset represent real-world challenging tasks due to stochastic fluctuations typical of sunspots and wind energy data. Both the quantum models have superior performances compared to the classical ones in all the test cases, and the QGRU outperforms the QLSTM despite having a simpler architecture.

The original contributions of our work can be summarized as follows:

- Our paper introduces, for the first time, a novel QGRU model composed of parameterized quantum circuits. The QGRU is distinct from existing models such as QLSTM, representing a novel approach to quantum RNN;
- We perform a comprehensive complexity analysis, demonstrating that the QGRU requires 25% fewer quantum parameters compared to the previously proposed QLSTM. This reduction in parameters is a significant advancement in quantum circuit design, contributing to the efficiency of quantum models;
- To the best of our knowledge, our work is the first to benchmark both QGRU and QLSTM on real-world challenging problems such as sunspots cycle and wind power generation forecasting. This application of quantum models to practical scenarios extends the scope of QNNs research.
- In all tested scenarios, our QGRU consistently outperforms QLSTM and their classical counterparts. This empirical evidence underscores the efficacy of our proposed model in solving real-world problems, demonstrating its practical utility.

- We empirically show that the QGRU network is approximately 25% faster in both the training and inference stages compared to QLSTM. This efficiency gain is crucial for making quantum algorithms more accessible and less demanding on current quantum devices and simulators.

This paper is organized as follows. Section II gives a detailed overview of the state-of-the-art for QNNs. In section III we introduce some basic concepts on Quantum Variational Learning. In section IV the architecture of the QGRU is described. Section V contains the experimental results of our work. Finally, the conclusions and future prospects are reported in section VI.

2. Related works

The quest for low-depth Quantum Machine Learning (QML) algorithms workable for small-scale quantum systems led to the adoption of Variational Quantum Circuits (VQCs), which currently represent the most effective way to harness quantum advantage offered by quantum computers [20, 21]. VQCs are based on a hybrid quantum–classical procedure: the quantum circuit is made up of parametrized quantum gates which can be optimized in an iterative framework with the help of a classical co-processor. VQCs allow to efficiently design low-depth QNNs that can be implemented on the available NISQ devices; the learnable parameters can effectively absorb noise during the iterative optimization process without incorporating any information coming from the noisy component. A layered architecture is used to encode input data into the circuit and process quantum states in a high-dimensional Hilbert space [22]. Such VQCs are succeeding in many common DL tasks [23, 24] as well as real-world applications [25]. For instance, authors in [26] propose a circuit-centric quantum classifier architecture which is ready to be implemented on NISQ devices. It predicts class labels of quantum encoded data via measurements of a chosen observable and demonstrates good resilience to noise. Researchers in [27] introduce a multi-layer Quantum Deep Neural Network (QDNN) with three variational layers to solve an image classification task. They found out that the QDNN have higher expressive power than classical Deep Neural Networks (DNN). An hybrid quantum–classical Convolutional Neural Network is presented in [28] as image classifier. It is employed in a remote sensing context, achieving better performance over the classical counterpart.

Nevertheless, despite being a frequently studied problem in DL [4, 29–31], time series forecasting has not been thoroughly investigated in the quantum realm. Although several quantum approaches were considered for time series prediction [12, 32–34], few works concerning quantum RNN have already demonstrated a concrete quantum advantage with respect to classical RNN in the literature. Researchers in [35] propose a Quantum Weighted Gated Recurrent Unit Neural Network (QWGRUNN) which shows improved performance in terms of prediction error compared to classical alternatives. Original GRU weights and activity values are substituted by weight-qubits and activity-qubits respectively, but this leads to increased computational complexity, making such an architecture unfeasible for current NISQ devices. Similarly [36], use a quantum weighted neuron to build a Mogrifier-Quantum Weighted Memory Enhancement LSTM (Mogrifier-QWMELSTM) to successfully forecast integrated energy distribution system time series. A Continuous-Variable Quantum RNN (CV-QRNN) network based on the continuous-variable quantum computing paradigm is introduced in [37], leading to faster convergence and better performance compared to the classical counterpart. However, it makes use of specialized quantum-photonic hardware, which may pose challenges in terms of experimental implementation and resource overheads with current technologies. [38] proposes a Parametrized Quantum LSTM (PQ-LSTM) model, i.e. a neural network which has an initial classical LSTM layer followed by a parametrized quantum layer. The PQ-LSTM is used to predict stress levels of knowledge workers, but the goodness of the model cannot be properly evaluated because no comparisons were made with the classical counterpart. Authors in [39] realize a Duplication-free Quantum LSTM (DQLSTM) neural network for natural language processing problems. However, the DQLSTM model encodes classical data into the quantum circuit through an amplitude encoding method, which is highly inefficient and unfeasible for large datasets. Both [40] and [19] propose an angle encoded QLSTM, where the underlying idea is to replace the learnable parameters W_* and b_* of the classical LSTM with variational layers composed of parametrized quantum gates. In the latter approach, VQCs layers are interposed between two classical Fully Connected (FC) layers to properly match the dimensionality of the input and output vectors, respectively. Based on the model proposed in [19], several simple applications in natural language processing [41, 42] and material synthesis [43] are present in the literature.

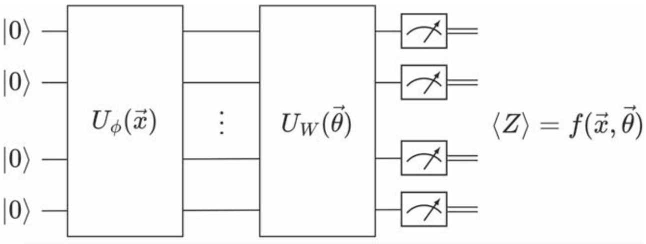

Figure 2. High-level scheme of a VQC with measurements in Pauli-Z basis.

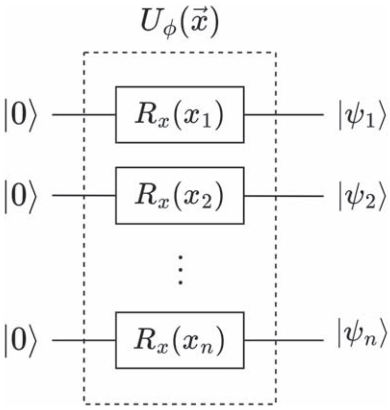

Figure 3. The chosen quantum feature map. Each feature value x_i, $i = 1...n$, is the argument of a parametrized R_x rotation gate.

3. Basic concepts on variational quantum circuits

VQCs represent the most promising approach to implement QML algorithms and QNNs on NISQ devices. A VQC consists of a data encoding stage, a properly designed ansatz applied to the quantum state and a measurement operation at the end of the circuit, as illustrated in figure 2.

To encode classical data $\vec{x} \in \mathrm{R}^n$ into an n qubit quantum circuit, a quantum feature map ϕ: $\mathrm{R}^n \rightarrow H^{2^n}$ is applied, where H^{2^n} is a 2^n dimensional Hilbert space. It corresponds to applying a unitary matrix $U_\phi(\vec{x})$ to the initial state $|0\rangle^{\otimes n}$:

$$U_\phi(\vec{x})|0\rangle^{\otimes n} = |\phi(\vec{x})\rangle = |\psi\rangle. \tag{1}$$

A suitable data encoding strategy is crucial for the attainment of a quantum advantage. An efficient representation of classical input data through quantum states is a necessary prerequisite to benefit from quantum technologies [44], and the selection of the unitary $U_\phi(\vec{x})$ has a considerable impact on the performance of the underlying VQC. Although various data encoding techniques have been proposed in the literature [22, 45], angle encoding is the most popular and efficient method to encode continuous variables into quantum states [15, 27, 44]. It employs parametrized rotation quantum gates to encode each input feature into a qubit: angle encoding allows to represent n input features by means of n qubits. The value of the rotation parameters directly corresponds to the value of the input features. In this paper, the chosen encoding consists of a layer of R_x parametrized gates, as shown in figure 3.

After data encoding, an ansatz $U_W(\vec{\theta})$ of $\vec{\theta}$-parametrized unitaries is applied to the state $|\psi\rangle$. Such unitaries are made of tunable single-qubit rotation gates, which are randomly initialized, and fixed entangling gates. Rotations are used to adequately control the quantum state space, while entanglement allows to create deeply correlated quantum vectors among all the qubits. During the computation, parameters optimization is performed by means of a classical co-processor in an iterative framework. Rotations and entanglement layers may be applied repeatedly to realize more expressive models [15]. The list of proposed ansatzes in literature is large and widely studied [15, 46]. The hardware-efficient ansatz [47], consisting of alternating layers of parametrized single-qubit gates and entangling two-qubits gates, has imposed as a promising solution thanks to the good trade-off between circuit expressivity and trainability [27, 28]. Hardware-efficient ansatzes have structured architectures, easy to be efficiently implemented on NISQ devices. For this reason, the ansatz selected in this work is made up of alternating layers of parametrized R_x gates followed by CNOT gates with circular

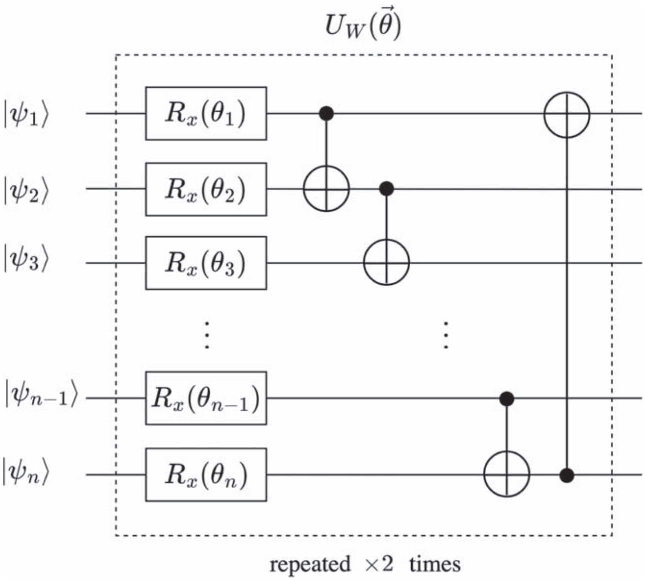

Figure 4. The design of the chosen ansatz with depth 2.

entanglement, as illustrated in figure 4. Circular entanglement connects every qubit with its neighbor in a sequential fashion; the last qubit is considered as a neighbor to the first qubit. To increase the generalization ability of the VQC, we chose to fix the ansatz depth to 2; a depth greater than 2 was not taken into consideration to prevent overfitting.

A measurement operation is performed at the end of the circuit to extract the outcome of the quantum circuit. Qubits are measured in a desired basis to obtain the following prediction:

$$f(\vec{x},\, \vec{\theta}) = \langle \phi(\vec{x})|\; U_W(\vec{\theta})^{\dagger} \hat{O} U_W(\vec{\theta}) |\phi(\vec{x})\rangle, \tag{2}$$

which corresponds to estimating the expectation value $\langle \hat{O} \rangle$ of an observable $\hat{O}$. Due to the probabilistic nature of quantum mechanics, several shots of the circuits are necessary to get the expected value of the designated observable. The expected value $\langle \hat{O} \rangle$ can be intended as the weighted sum of the eigenvalues, where the weights represent the probabilities that the measured state vector is in the associated eigenstate. For this work, measurements are carried out in the Pauli-Z basis, which has $+1$ and -1 as eigenvalues. Consequently, the output of the VQC for a given qubit state will be restricted to the $[-1, 1]$ range.

The outcome $f(\vec{x},\, \vec{\theta})$ of the VQC is then evaluated into a cost function $C(f(\vec{x},\, \vec{\theta}))$ and the parameters $\vec{\theta}$ are properly optimized using a classical processor. Such quantum and classical steps are run iteratively in cycle to find better $\vec{\theta}$ parameters at every step. To update $\vec{\theta}$ and train the QNNs, gradient-based techniques can be used; gradients in a parametrized quantum circuit are calculated via the parameter-shift rule:

$$\nabla_{\theta} f(x,\, \theta) = \frac{1}{2}\left[f\left(x,\, \theta + \frac{\pi}{2}\right) - f\left(x,\, \theta - \frac{\pi}{2}\right)\right], \tag{3}$$

where $f(\vec{x},\, \theta)$ is the output of the quantum circuit and θ is the parameter to be optimized.

4. Quantum GRU

Here we propose QGRU, a novel recurrent QNN architecture which extends the GRU model by introducing VQCs layers into each cell. For the sake of completeness, the equations governing the behavior of a GRU cell are hereafter reported:

$$r_t = \sigma(W_r[h_{t-1},\, x_t] + b_r), \tag{4}$$

$$z_t = \sigma(W_z[h_{t-1},\, x_t] + b_z), \tag{5}$$

$$\tilde{h}_t = \tanh(W_o[r_t \times h_{t-1},\, x_t] + b_n), \tag{6}$$

$$h_t = (1 - z_t) \times \tilde{h}_t + z_t \times h_{t-1}. \tag{7}$$

Once again, symbol '×' denotes element-wise (Hadamard) multiplication.

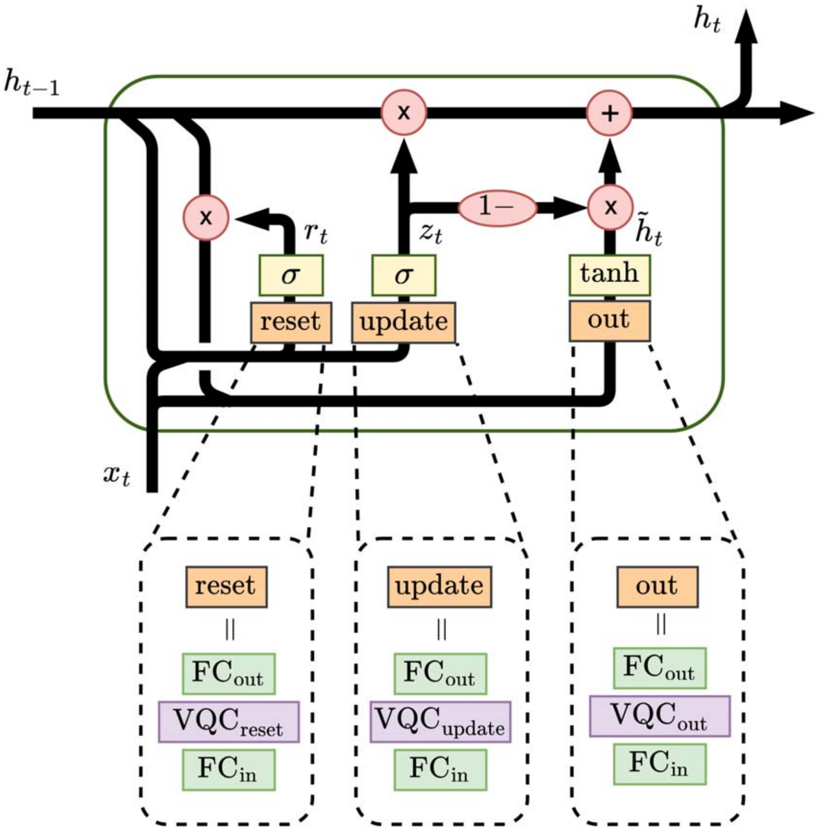

Figure 5. Internal scheme of a QGRU cell.

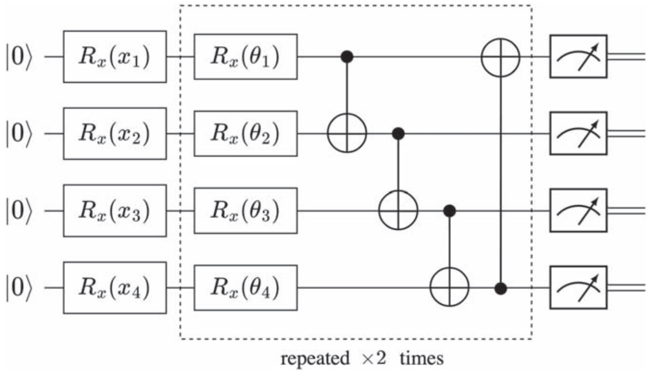

Figure 6. Circuital schema of the proposed VQC.

4.1. Architecture of a QGRU cell

The rationale is to substitute the original GRU's weight matrices with VQCs and FC layers to leverage quantum properties and improve the computation. In total, a QGRU cell contains 3 different VQCs layers, one per each nonlinear GRU function, as depicted in figure 5. The VQCs allow to effectively project input data into a high-dimensional feature space and apply unitary transformations to an exponential state space, thus enhancing information processing inside the cell. Every VQC layer is interspersed with two classical FC layers, namely $\mathrm{FC_{in}}$ and $\mathrm{FC_{out}}$, whose role is to match the input and output dimensionality of the quantum layer respectively. In order to reduce the number of classical parameters and lighten the computation, the weights of the $\mathrm{FC_{in}}$ and $\mathrm{FC_{out}}$ layers are the same for every cell gate. The design of a VQC layer is reported in figure 6. It is composed of a layer of R_x gates for data encoding, an ansatz of parametrized R_x gates with circular CNOT entanglement and final measurement operations applied to all the qubits. The equations describing the behavior of a QGRU cell are hereafter listed:

$$r_t = \sigma(\mathrm{FC_{out}}(\mathrm{VQC_{reset}}(\mathrm{FC_{in}}([h_{t-1}, x_t])))), \tag{8}$$

$$z_t = \sigma(\mathrm{FC}_{\mathrm{out}}(\mathrm{VQC}_{\mathrm{update}}(\mathrm{FC}_{\mathrm{in}}([h_{t-1}, x_t])))), \tag{9}$$

$$\tilde{h}_t = \tanh(\mathrm{FC}_{\mathrm{out}}(\mathrm{VQC}_{\mathrm{out}}(\mathrm{FC}_{\mathrm{in}}([r_t \times h_{t-1}, x_t])))), \tag{10}$$

$$h_t = (1 - z_t) \times \tilde{h}_t + z_t \times h_{t-1}, \tag{11}$$

where the $\mathrm{VQC}_*(\cdot)$ function represents the processing inside the quantum circuit. The output of all the variational layers is an n-dimensional vector corresponding to Pauli-Z expectation values of each qubit, where n is the number of qubits in the quantum circuit. This way, it is possible to apply nonlinear functions such as $\sigma(\cdot)$ and $\tanh(\cdot)$ to the output of the quantum circuit, introducing nonlinearity at the end of the quantum computation. The selection of nonlinear functions, specifically the sigmoid $\sigma(\cdot)$ and hyperbolic tangent $\tanh(\cdot)$ functions, is based on their well-established roles in the architecture of GRU cells. The sigmoid function $\sigma(\cdot)$ is chosen for its ability to map input values to a range between 0 and 1. This property makes it particularly useful for gating mechanisms within the GRU, such as the update and reset gates, where it effectively controls the extent to which information is updated or forgotten. The output of the sigmoid function can be interpreted as a probability or a level of activation, which is crucial for regulating the flow of information through the network. The hyperbolic tangent function $\tanh(\cdot)$ is selected for its capability to map input values to a range between -1 and 1. This characteristic makes it ideal for creating a balanced output that includes both positive and negative values, which is beneficial for learning complex patterns in the data and providing a range that supports gradient propagation and stable learning dynamics. Regarding the sensitivity of the procedure to these design choices, it is important to note that the performance of GRU cells can indeed be influenced by the choice of activation functions. While $\sigma(\cdot)$ and $\tanh(\cdot)$ are commonly used due to their effectiveness in practice, alternative activation functions can be explored. However, deviations from these standard functions may require additional tuning of other hyperparameters and careful consideration of the overall network architecture to maintain stability and performance.

4.2. Information processing inside a QGRU cell

The information flow inside each QGRU cell is hereafter explained. First, x_t and h_{t-1} input data are concatenated into the input vector $[h_{t-1}, x_t]$. The latter is then fed into the reset block and the update block, where $\mathrm{VQC}_{\mathrm{reset}}$ and $\mathrm{VQC}_{\mathrm{update}}$ are used to delete and manipulate information respectively. The resulting r_t multiplies the past hidden state h_{t-1}; the vector $[r_t \times h_{t-1}, x_t]$ is processed by the classical and the $\mathrm{VQC}_{\mathrm{out}}$ layers to calculate the candidate hidden state $\tilde{h}_t$. Finally, the new hidden state h_t is obtained as a result of the output block computation, involving z_t, the candidate hidden state $\tilde{h}_t$ and the past hidden state h_{t-1}.

Such QGRU architecture based on VQCs and classical FC layers provides great flexibility in the construction of the network. The VQCs layers can be employed to better approximate the complex dynamics of the underlying time series data without worrying about their dimensionality, which is handled by the FC layers. The parameters of the model are properly updated via the backpropagation algorithm. The parameter-shift rule is used to compute the gradient of the quantum circuits, which is then backpropagated through the network.

4.3. Complexity analysis

Finally, it is worth to conduct a complexity analysis of both the QLSTM and the QGRU models; the equations governing the behavior of the QLSTM can be found in [40]. Let d_{hid} be the hidden dimension, n the number of qubits composing each quantum circuit, d_{in} the number of input features and let l be the number of ansatz layers of each VQC. The number of parameters composing a single VQC layer is $n\ \ l$. The $\mathrm{FC}_{\mathrm{in}}$ classical layer receives d_{conc} features in input, where d_{conc} is the dimension of the concatenated vectors $[h_{t-1}, x_t]$ and $[r_t \times h_{t-1}, x_t]$, i.e. $d_{\mathrm{conc}} = d_{\mathrm{hid}} + d_{\mathrm{in}}$, and outputs a vector with n dimensions. Similarly, the $\mathrm{FC}_{\mathrm{out}}$ layer receives in input n features and outputs a vector with d_{hid} features. Therefore, the number of parameters in the $\mathrm{FC}_{\mathrm{in}}$ and $\mathrm{FC}_{\mathrm{out}}$ layers are $nd_{\mathrm{conc}} + n$ and $nd_{\mathrm{hid}} + d_{\mathrm{hid}}$ respectively. The QLSTM is composed of 3 gates and 1 cell buffer, so it has 4 VQCs, 1 $\mathrm{FC}_{\mathrm{in}}$ and 1 $\mathrm{FC}_{\mathrm{out}}$ trainable layers. Then, the total number of parameters composing a QLSTM cell is $4nl + (nd_{\mathrm{conc}} + n) + (nd_{\mathrm{hid}} + d_{\mathrm{hid}})$, which is equal to $n(4l + 2d_{\mathrm{hid}} + d_{\mathrm{in}} + 1) + d_{\mathrm{hid}}$; the number of quantum parameters is $4nl$. Similarly, a QGRU cell has 3 VQCs, 1 $\mathrm{FC}_{\mathrm{in}}$ and 1 $\mathrm{FC}_{\mathrm{out}}$ trainable layers. The number of parameters in a QGRU cell is $n(3l + 2d_{\mathrm{hid}} + d_{\mathrm{in}} + 1) + d_{\mathrm{hid}}$, where only $3nl$ are quantum parameters. Consequently, it is evident that QGRU uses 25% quantum parameters less than QLSTM, leading to faster computation. This is particularly suitable for the current state of quantum technology, since both quantum simulators and real devices struggle in executing quantum circuits efficiently.

5. Experimental validation

We performed three different experiments to assess the goodness of the QGRU model. The first experiment, which we called Experiment I, consisted in learning a simple periodic function, which is a common benchmark for recurrent networks. The second experiment, namely Experiment II, tested the performance of the model on

Table 1. Network structures and related hyperparameters.

Model	d_{hid}	qubits	l
QGRU	3	4	2
QLSTM	3	4	2
GRU	3	—	—
LSTM	3	—	—

the monthly average number of sunspots from January 1749 to July 2018. Predicting sunspots is a widely studied yet not trivial problem in physics due to their fluctuations and recurrent neural models are usually employed to obtain satisfactory results, so a performance evaluation of QGRU in this context was appropriate. Finally, in Experiment III we tackled a challenging real-world problem related to wind power production time series forecasting. The network was trained to predict actual wind output power generated in Italy in the month of June 2022. The prediction of wind energy data is considered an ambitious task because the underlying time series may exhibit a very unstable behavior, depending on seasonality as well as on weather conditions.

These datasets were selected due to their established use as benchmarks for evaluating recurrent networks and their intrinsic characteristics, which pose varying degrees of complexity in time series prediction. The periodic function dataset, used in Experiment I, is a standard benchmark to assess the ability of models to learn and predict periodic patterns effectively. This choice is not inherently tied to the capabilities of the basic model but rather serves to evaluate the fundamental competency of the model in capturing and predicting regular periodicity. In Experiment II, the sunspot dataset is known for its challenging nature due to the stochastic fluctuations and inherent periodicity of sunspot activity. This dataset is frequently employed in literature to test the robustness of recurrent models against complex real-world time series with quasi-periodic characteristics. For Experiment III, the wind power generation dataset represents a real-world application with highly unstable behavior influenced by weather conditions and seasonality. The choice of this dataset aims at testing the model's ability to handle highly volatile and non-periodic data, further demonstrating its generalization capabilities beyond periodic or quasi-periodic data.

5.1. Experimental setup

In all the experiments, the forecasting problem was modeled as follows. Given the scalar time series of interest S and a window size k, all the observations from the current sample t up to the sample $t - k + 1$ are fed into the model. The latter is trained to forecast the next sample $t + 1$, thus resulting in the forecasted scalar $\tilde{S}[t + 1]$. The window size was chosen to be $k = 5$ for all the test cases.

We compared our QGRU with a QLSTM and with their classical counterparts, that is a GRU and an LSTM. We also compared all the models with a Naive predictor, which is a straightforward forecasting method where the prediction for any given time step is simply the value observed in the previous time step. Formally, for a time series S, the Naive predictor is defined such that for each value $S[t]$ at step t, the prediction for the next step $t + 1$ can be expressed as:

$$\tilde{S}[t + 1] = S[t]. \tag{12}$$

This straightforward approach serves as a baseline reference for evaluating the performance of more sophisticated models and allows us to assess whether the proposed quantum-inspired architectures outperform a simple heuristic approach. The Naive predictor's simplicity, minimal assumptions, and established use in literature [48–50] make it an appropriate and effective baseline for evaluating the performance of more advanced models. Since it operates under the reasonable assumption that future values are similar to the most recent observed value, it provides a basic performance standard that any advanced model should exceed. The list of hyperparameters of the networks, which is the same for all the experiments, is reported in table 1, where d_{hid} is the hidden dimension of the recurrent cell, 'qubits' indicates the number of qubits and l the number of layers inside each VQC. The hyperparameters of the network were selected after a rigorous grid search procedure, bearing in mind that as the number of qubits and quantum layers increased, the computation became exponentially expensive. We noticed that different hyperparameters led to similar outcomes and did not affect significantly the results of the experiments. To make a fair comparison, the magnitude of parameters in each quantum model was chosen in such a way that it was comparable with the number of parameters inside the respective classical models. The total number of parameters for each recurrent model is presented in table 2.

Moreover, all the networks had a final FC layer to compile the output of the recurrent layer to a suitable output dimension, i.e. a scalar output.

In each experiment, the corresponding dataset was divided into 80% train and 20% test. Moreover, before the training phase, the raw time series was scaled in the range $[-1, 1]$ to best suit the output of the VQCs; the

Table 2. Number of classical and quantum parameters.

Model	Classical	Quantum	Total
QGRU	35	24	59
QLSTM	35	32	67
GRU	54	—	54
LSTM	72	—	72

Table 3. Training hyperparameters.

Experiment	Epochs	Initial learning rate	learning rate drop period	learning rate drop factor
I	130	0.01	95	0.7
II	120	0.01	100	0.7
III	180	0.01	160	0.8

Table 4. Average MSE and MAE with standard deviation for experiment i test set.

Model	MSE	MAE
QGRU	0.00144 ± 0.00086	0.03269 ± 0.01152
QLSTM	0.00214 ± 0.00056	0.04215 ± 0.00573
GRU	0.00268 ± 0.00173	0.04396 ± 0.01497
LSTM	0.00233 ± 0.00138	0.04135 ± 0.01313
Naive	0.00487	0.06245

scaler was fitted using the training set only. To guarantee a good and fast convergence, the models were trained using the RMSprop algorithm and an extensive grid search procedure was carried out for the tuning of training hyperparameters, which are listed in table 3 for each experiment.

We have empirically tested various hyperparameters and optimizers to determine the most effective training method for our models. Our experiments revealed that RMSprop slightly outperformed ADAM in terms of convergence speed and overall model performance. These findings are in alignment with recent literature on QNNs [40, 51–53], which suggests that RMSprop can be more effective than other optimizers in navigating certain QNN loss landscapes.

All the networks were implemented in Python 3.8 with PyTorch. The quantum layers were realized in PennyLane, a framework which enables local quantum circuits simulations and integration with classical NN. Since we used a Python-based simulator to run our quantum circuits, we adopted PennyLane's adjoint differentiation method [54] to speed up the computation, which guarantees a fast and memory efficient way to differentiate quantum circuits. A machine equipped with an AMD Ryzen 7$^{\mathrm{TM}}$ 5800X 8-Core CPU at 3.80 GHz and with 64 GB of RAM was used for the experiments.

The Mean Squared Error (MSE) and Mean Absolute Error (MAE) were selected as error metrics to evaluate the performances of the models, as they are standard error metrics in supervised learning. MSE was also selected as loss function to train the networks because it is more sensitive to larger errors. In order to prove the robustness of our approach against random parameters initializations, we performed 10 runs of the algorithms for every test case.

5.2. Experiment I: periodic function

As first experiment, we investigated QGRU's ability in learning a simple periodic function, that is the sine function $y = \sin(x)$. Despite it represents a basic learning task, it is commonly used as a reference benchmark for recurrent neural networks [40, 55]. As shown in table 4, the results of Experiment I highlight that the QGRU model totally outperformed the other networks as for the error rate, both in terms of MSE and MAE. In particular, the MSE for the QGRU is 38% less than the one obtained by the LSTM, 46% less than the one obtained by the GRU and 33% less than the MSE obtained by the QLSTM. At the same time, the QLSTM performed better than the two classical models: its MSE error is 8% and 20% less than the one achieved by LSTM and GRU, respectively. In addition, QGRU and QLSTM's errors are more stable than GRU and LSTM's, as

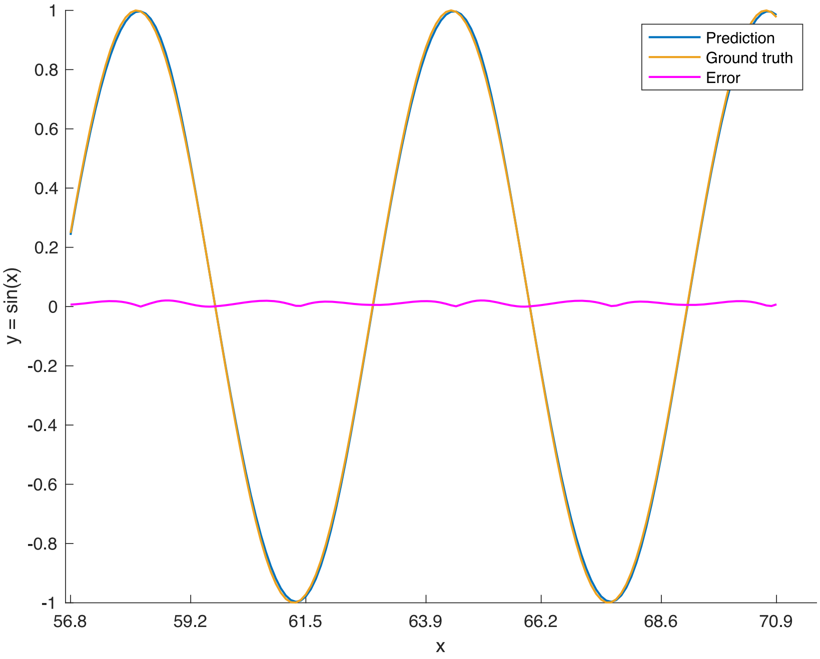

Figure 7. Comparison between the QGRU (blue) and the ground truth (orange) for Experiment I test case. The absolute error between the prediction and the ground truth (magenta) is also reported to clearly highlight the differences between the two curves.

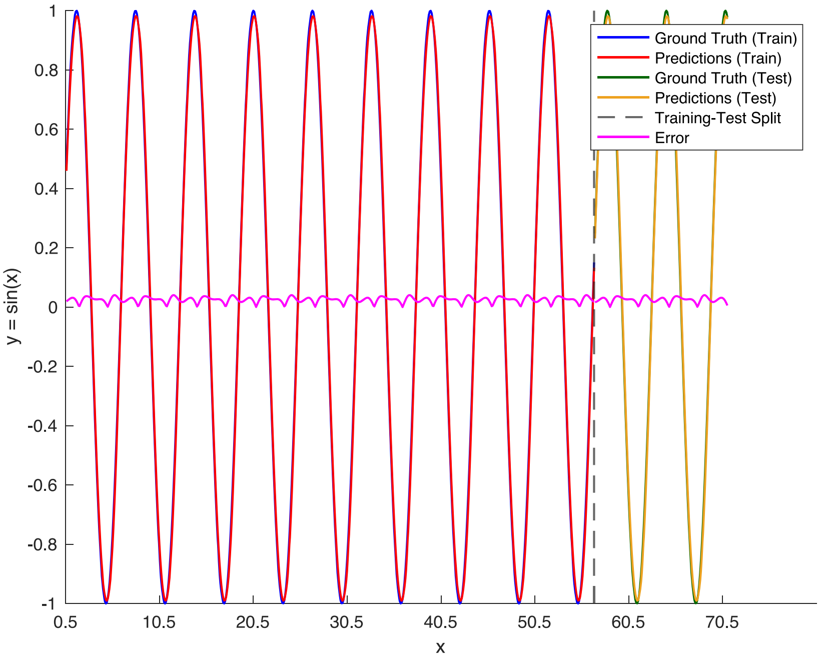

Figure 8. Comparison between the QGRU training predictions (red), training ground truth (blue), QGRU test predictions (orange) and test ground truth (green) for Experiment I test case. The black dashed line represents the 80%–20% Training-Test data split. The absolute error between the prediction and the ground truth (magenta) is also reported to clearly highlight the differences between the two curves.

indicated by the lower standard deviation of such errors. All the neural models outperformed the Naive predictor, which has an almost doubled MSE and MAE compared to them.

The good behavior of the QGRU is illustrated in figures 7 and 8, where it is evident that the predicted curves strictly follow the original underlying functions. We can observe larger errors near the turning points of the sinusoidal function, precisely just before and after the turning points. This behavior is primarily due to the model's limited complexity, as it consists of only one layer. This single-layer architecture restricts the model's

Table 5. Average MSE and MAE with standard deviation for experiment II test set.

Model	MSE	MAE
QGRU	540.47 ± 6.76	17.18 ± 0.17
QLSTM	545.94 ± 23.63	17.20 ± 0.33
GRU	556.07 ± 29.57	17.63 ± 0.84
LSTM	545.62 ± 11.82	17.31 ± 0.41
Naive	636.01	18.20

ability to capture the sharp changes in the function's direction at the peaks and troughs effectively. Consequently, the model struggles to approximate the steep gradients and rapid transitions accurately, resulting in slightly higher errors in these regions.

5.3. Experiment II: sunspots cycle

In this second experiment, we studied the capability of the proposed QGRU in approximating a real-world sequence, i.e. the monthly mean total sunspot number from January 1749 to July 2018. Sunspots are temporary phenomena manifesting on the Sun's photosphere as areas darker than their surroundings. Data was gathered together by the solar physics research department of the Royal Observatory of Belgium. Due to the intrinsic fluctuations typical of this phenomenon, it is considered a hard prediction task even for recurrent neural models. Nevertheless, QGRU still obtained satisfactory results, showcasing a reduction in MSE by 5.15, 15.6 and 5.47 points compared to the LSTM, GRU and QLSTM models respectively, as reported in table 5. The QLSTM network achieved comparable results with respect to the classical counterpart in terms of MSE, while achieving a 0.11 points lower MAE. Both quantum models greatly surpassed GRU's performance in MSE and MAE metrics. The robustness of the proposed QGRU model was substantiated through an examination of standard deviation values. Specifically, the standard deviation was twice as high in the conventional LSTM architecture and even four times greater in QLSTM and GRU networks when contrasted with the values observed in the QGRU model. In general, the underlying intuition is that quantum models were able to project input data into a high-dimensional quantum feature space, thus finding new hidden patterns among temporal observations; the resilience against stochastic fluctuations made QGRU and QLSTM able to conduct a more effective and noise-resistant data processing.

Despite the aforementioned difficulty in predicting stochastic temporal sequences, all the neural models greatly outperformed the Naive predictor and found an underlying pattern in sunspots data. Also in this setting, QGRU's prediction still demonstrated to be very accurate compared to the ground truth time series, as shown in figure 9 and figure 10.

5.4. Experiment III: wind power generation

Finally, Experiment III assessed the performance of the proposed quantum model in another real-world scenario: the task was to forecast the wind power generation, measured in GWh, occurred in Italy in the month of June 2022. Data were sampled hourly, thus the dataset was composed of 24 daily observations per 30 days, resulting in 720 total samples. Due to wind's highly unstable behavior, caused by weather conditions as well as by seasonality, wind power prediction is considered to be an exceptionally hard task. Again, the rationale here was to exploit the quantum model's ability to approximate highly complex and volatile sequences thanks to its high-dimensional quantum state space. The performances of all the models are reported in table 6. Except for the LSTM, all the neural models greatly outperformed the Naive approach, showcasing their capacity to capture intricate patterns and dependencies within the time series data. Despite being a simple network, QGRU had remarkable performances: the MSE error with respect to LSTM, GRU and QLSTM decreased by 40%, 28% and 4% respectively. Similarly, the MSE for the QLSTM was 37% lower than the one for the LSTM and 25% lower than the error obtained by using the GRU. Alike the previous experiments, also in this case the standard deviation revealed the robustness of the proposed approach, being less than half the standard deviation of the classical counterparts.

A visual comparison between QGRU's predictions and the observed sequence is illustrated in figure 11 and in figure 12. The quantum model accurately predicted the underlying time series: the predicted sequence overlaps almost perfectly with the true one, and also the fluctuating and spiking behavior was modeled correctly.

5.5. Training and inference times

Ultimately, another significant result that we want to stress in this paper is related to QGRU's runtime performances, which were expected to be about 25% better than the QLSTM ones, since the quantum layers act

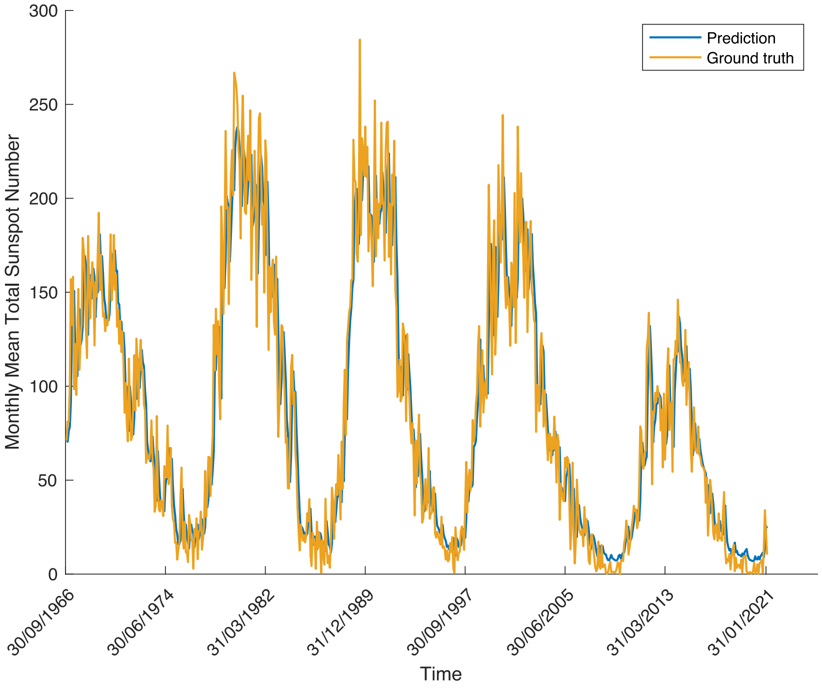

Figure 9. Comparison between the QGRU (blue) and the ground truth (orange) for Experiment II test case.

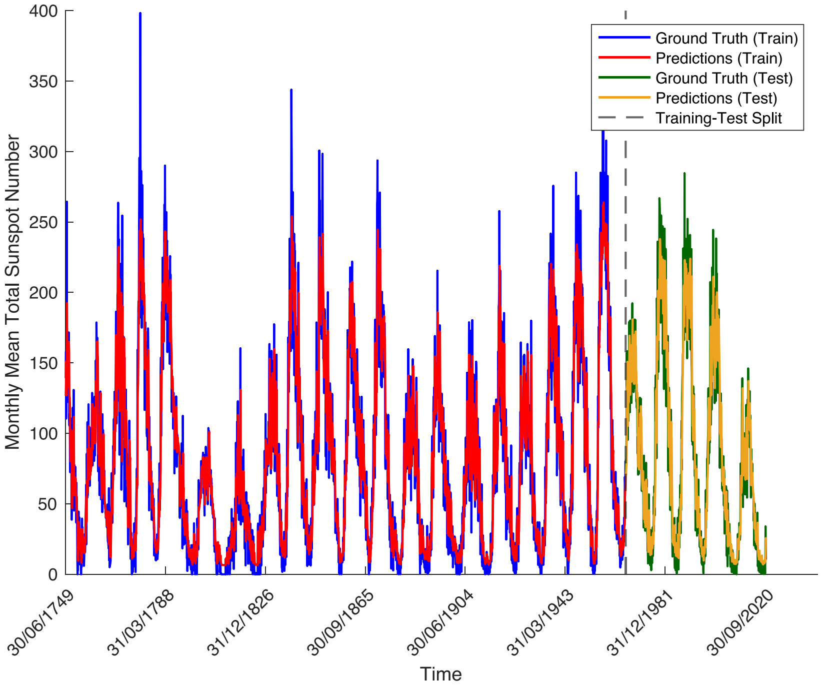

Figure 10. Comparison between the QGRU training predictions (red), training ground truth (blue), QGRU test predictions (orange) and test ground truth (green) for Experiment II test case. The black dashed line represents the 80%–20% Training-Test data split.

as bottlenecks in the computation. In order to compare them, we calculated the average training and inference times of the two models for each experiment, as reported in table 7. The training times were computed by averaging on the entire training process over all the 10 runs, while the inference times correspond to the individual sample (i.e. time series) processed by the algorithm. It is worth to highlight that the reported runtime performances pertain to executions carried out on the Python simulator and not on a real device. Although the training and inference times would be drastically lower if computed on an actual quantum system, a faster quantum recurrent neural network architecture would be beneficial both in simulation and realistic settings.

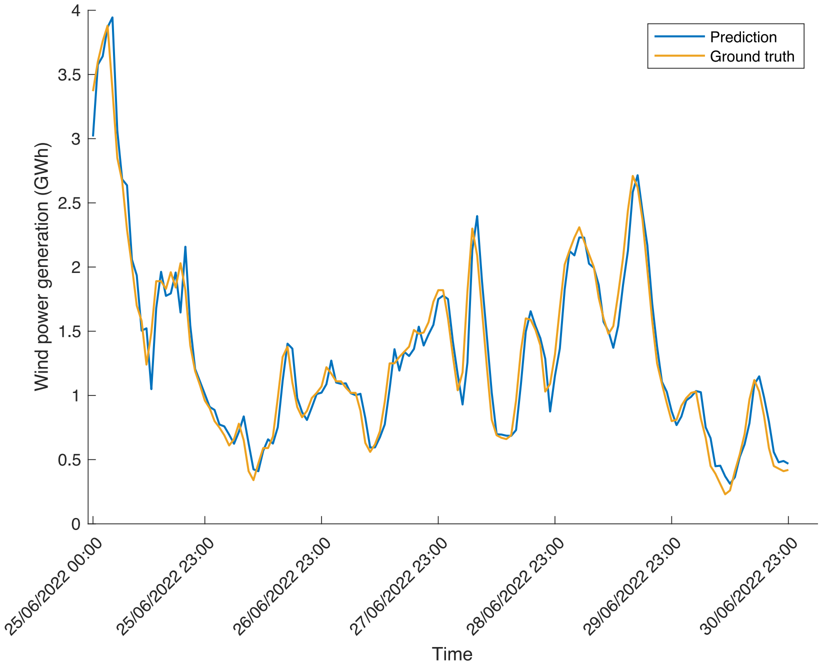

Figure 11. Comparison between the QGRU (blue) and the ground truth (orange) for Experiment III test case.

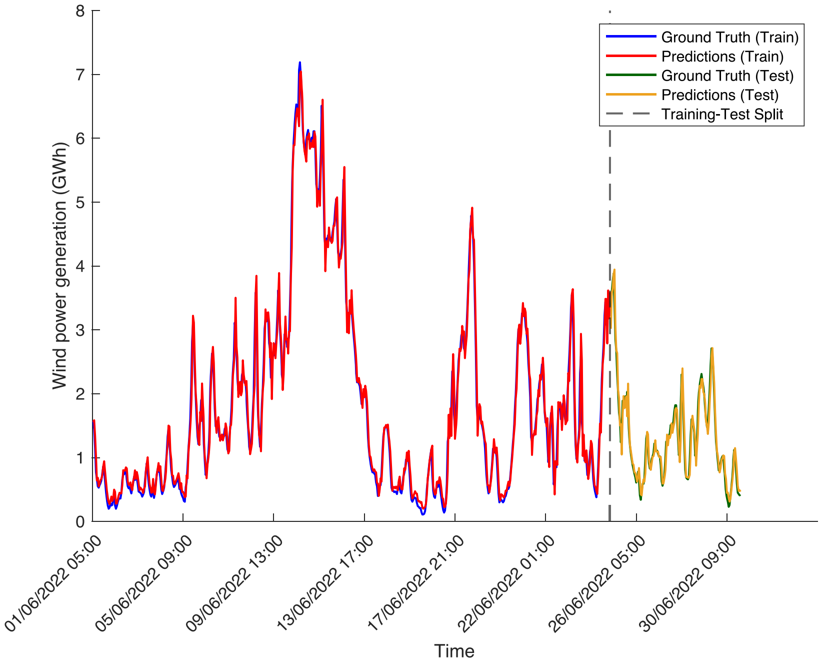

Figure 12. Comparison between the QGRU training predictions (red), training ground truth (blue), QGRU test predictions (orange) and test ground truth (green) for Experiment III test case. The black dashed line represents the 80%–20% Training-Test data split.

Another major aspect to point out is that QML models have not been optimized for GPU usage yet, hence the computational times of the quantum models refer to the CPU only.

The results presented in table 7 underline QGRU's remarkable runtime performances, which are about 25% faster than QLSTM's. They are in line with the expected theoretical results and confirm the efficiency of the proposed approach. For the sake of completeness, the average training and inference times of GRU and LSTM models were reported in table 7 as well. While it might seem informative to compare these times directly, it is important to note that simulations of quantum models on classical hardware are inherently slower than their classical counterparts. Comparing their average training and inference times could thus be misleading, as it does not accurately reflect the true performance potential of quantum models.

Table 6. Average MSE and MAE with standard deviation for experiment III test set.

Model	MSE	MAE
QGRU	0.02891 ± 0.00291	0.13009 ± 0.00838
QLSTM	0.03011 ± 0.00477	0.13290 ± 0.01186
GRU	0.04004 ± 0.00683	0.15402 ± 0.01444
LSTM	0.04815 ± 0.01072	0.16910 ± 0.01668
Naive	0.04358	0.16340

Table 7. Average training and inference times (s) of QGRU and QLSTM models for each experiment.

Experiment	Model	Training	Inference
I	QGRU	3.503×10^{3}	3.710×10^{-2}
	QLSTM	4.668×10^{3}	4.920×10^{-2}
	GRU	5.225×10^{-1}	4.800×10^{-6}
	LSTM	7.255×10^{-1}	6.943×10^{-6}
II	QGRU	2.017×10^{4}	2.900×10^{-2}
	QLSTM	2.887×10^{4}	3.820×10^{-2}
	GRU	1.592×10^{0}	1.675×10^{-6}
	LSTM	2.842×10^{0}	3.778×10^{-6}
III	QGRU	4.959×10^{3}	3.740×10^{-2}
	QLSTM	6.581×10^{3}	4.930×10^{-2}
	GRU	7.474×10^{-1}	3.018×10^{-6}
	LSTM	1.072×10^{0}	4.117×10^{-6}

6. Conclusions

In this paper, we proposed a novel QGRU architecture which is simpler, faster and more performant than the existing QLSTM network. To assess the goodness of the QGRU and QLSTM models, they were tested on a simple periodic function as well as on real-world challenging problems related to time series forecasting. To the best of our knowledge, it was the first time that quantum recurrent models were tested on real-world challenging problems such as sunspots and wind power generation forecasting. We provided an extensive fair comparison between the quantum models and the classical counterparts, highlighting the goodness of the QGRU over all the other models, as well as the goodness of the QLSTM over the two classical models. QGRU has also demonstrated to be faster than QLSTM, as evidenced by training and inference times comparisons among all the three experiments conducted on the simulator. The core idea at the basis of this paper is that quantum recurrent neural models may attain lower generalization errors over the classical ones due to the projection of input data into quantum high-dimensional states, which allows to capture hidden patterns among data and be more resilient to underlying data noise. As highlighted in previous works [56, 57], quantum entanglement facilitates inseparable associations among input timestep states, creating a comprehensive quantum probabilistic description of the sequence of states over time. This description is represented by a quantum pure state vector, which, when entangled, encodes the non-classical correlations between timesteps states, thereby enhancing the model's ability to capture intricate dependencies within the data.

As future works, the quantum networks could be run on a real quantum hardware to validate both the goodness of the results and the computational times on actual quantum machines. More complex data encoding strategy may be considered and an analysis on different kinds of ansatzes could be pursued as well. Finally, it would be worth to investigate the performances of quantum recurrent neural models on multivariate time series, considering their inherent ability in finding hidden correlations among features.

Acknowledgments

The contribution of all Authors in this work was supported by the 'NATIONAL CENTRE FOR HPC, BIG DATA AND QUANTUM COMPUTING' (CN1, Spoke 10) within the Italian 'Piano Nazionale di Ripresa e Resilienza

(PNRR)', Mission 4 Component 2 Investment 1.4 funded by the European Union-NextGenerationEU - CN00000013-CUP B83C22002940006.

Data availability statement

No new data were created or analysed in this study.

ORCID iDs

Andrea Ceschini https://orcid.org/0000-0003-0555-9543
Antonello Rosato https://orcid.org/0000-0002-4371-5925
Massimo Panella https://orcid.org/0000-0002-9876-1494

References

[1] Voulodimos A, Doulamis N, Doulamis A and Protopapadakis E 2018 Deep learning for computer vision: a brief review *Computational Intelligence and Neuroscience* **2018** 7068349

[2] Sutskever I, Vinyals O and Le Q V 2014 Sequence to sequence learning with neural networks *Proceedings of the 27th International Conference on Neural Information Processing Systems - NIPS∍14* vol 2 (MIT Press) 3104–12

[3] Joseph J, Vineetha S and Sobhana N 2022 A survey on deep learning based sentiment analysis *Materials Today: Proceedings* **58** 456–60

[4] Ceschini A, Rosato A, Succetti F, Luzio F D, Mitolo M, Araneo R and Panella M 2021 Deep neural networks for electric energy theft and anomaly detection in the distribution grid *2021 IEEE International Conference on Environment and Electrical Engineering and 2021 IEEE Industrial and Commercial Power Systems Europe (EEEIC/I&CPS Europe)* pp 1–5

[5] Hochreiter S and Schmidhuber J 1997 Long short-term memory *Neural Comput.* **9** 1735–80

[6] Yu Y, Si X, Hu C and Zhang J 2019 A review of recurrent neural networks: LSTM cells and network architectures *Neural Comput.* **31** 1235–70

[7] Cho K, van Merrienboer B, Bahdanau D and Bengio Y 2014 On the properties of neural machine translation: encoder-decoder approaches *Proceedings of {SSST}-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation* ed D Wu, M Carpuat, X Carreras and E M Vecchi (Association for Computational Linguistics) 103–11

[8] Chung J, Gulcehre C, Cho K and Bengio Y 2014 Empirical evaluation of gated recurrent neural networks on sequence modeling *NIPS 2014 Workshop on Deep Learning, December 2014*

[9] Yang S, Yu X and Zhou Y 2020 LSTM and GRU neural network performance comparison study: Taking yelp review dataset as an example *2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)* pp 98–101

[10] Fu R, Zhang Z and Li L 2016 Using LSTM and GRU neural network methods for traffic flow prediction *2016 XXXI Youth academic annual conference of Chinese association of automation (YAC). IEEE* pp 324–8

[11] Chen B-Q and Niu X-F 2020 A novel neural network based on quantum computing *Int. J. Theor. Phys.* **59** 2029–43

[12] Ceschini A, Rosato A and Panella M 2022 Design of an LSTM cell on a quantum hardware *IEEE Trans. Circuits Syst. Express Briefs* **69** 1822–6

[13] Panella M and Martinelli G 2009 Neurofuzzy networks with nonlinear quantum learning *IEEE Trans. Fuzzy Syst.* **17** 698–710

[14] Tacchino F, Macchiavello C, Gerace D and Bajoni D 2019 An artificial neuron implemented on an actual quantum processor *Npj Quantum Information* **5** 26

[15] Abbas A, Sutter D, Zoufal C, Lucchi A, Figalli A and Woerner S 2021 The power of quantum neural networks *Nature Computational Science* **1** 403–9

[16] Tacchino F, Barkoutsos P, Macchiavello C, Tavernelli I, Gerace D and Bajoni D 2020 Quantum implementation of an artificial feed-forward neural network *Quantum Science and Technology* **5** 044010

[17] Wan K H, Dahlsten O, Kristjánsson H, Gardner R and Kim M S 2017 Quantum generalisation of feedforward neural networks *Npj Quantum Information* **3** 36

[18] Huber P, Haber J, Barthel P, García-Ripoll J J, Torrontegui E and Wunderlich C 2021 Realization of a quantum perceptron gate with trapped ions arXiv: 2111.08977

[19] Di Sipio R, Huang J-H, Chen S Y-C, Mangini S and Worring M 2022 The dawn of quantum natural language processing *ICASSP 2022-2022 IEEE International Conference on Acoustics* pp 8612–6

[20] Mitarai K, Negoro M, Kitagawa M and Fujii K 2018 Quantum circuit learning *Phys. Rev.* A **98** 032309

[21] Tacchino F, Barkoutsos P K, Macchiavello C, Gerace D, Tavernelli I and Bajoni D 2020 Variational learning for quantum artificial neural networks *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)* pp 130–6

[22] Havlíček V, Córcoles A D, Temme K, Harrow A, Kandala A, Chow J and Gambetta J 2019 Supervised learning with quantum-enhanced feature spaces *Nature* **567** 209–12

[23] Dallaire-Demers P-L and Killoran N 2018 Quantum generative adversarial networks *Phys. Rev.* A **98** 012324

[24] Trochun Y, Stirenko S, Rokovyi O, Alienin O, Pavlov E and Gordienko Y 2021 Hybrid classic-quantum neural networks for image classification *2021 XI IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)* p 2

[25] Mangini S, Marruzzo A, Piantanida M, Gerace D, Bajoni D and Macchiavello C 2022 Quantum neural network autoencoder and classifier applied to an industrial case study *Quantum Machine Intelligence* **4** 1–13

[26] Schuld M, Bocharov A, Svore K M and Wiebe N 2020 Circuit-centric quantum classifiers *Phys. Rev.* A **101** 032308

[27] Zhao C and Gao X-S 2021 QDNN: Deep neural networks with quantum layers *Quantum Machine Intelligence* **3** 15

[28] Sebastianelli A, Zaidenberg D A, Spiller D, Saux B L and Ullo S L 2022 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** 565–80

[29] Torres J F, Hadjout D, Sebaa A, Martínez-Álvarez F and Troncoso A 2021 Deep learning for time series forecasting: a survey *Big Data* **9** 3–21

[30] Succetti F, Ceschini A, Luzio F D, Rosato A and Panella M 2021 Time series prediction with autoencoding LSTM networks, in International Work-Conference on Artificial *Neural Networks* (Springer) pp 306–17

[31] Succetti F, Rosato A and Panella M 2023 An adaptive embedding procedure for time series forecasting with deep neural networks *Neural Netw.* **167** 715–29

[32] Ueguchi T, Matsui N and Isokawa T 2016 Chaotic time series prediction by qubit neural network with complex-valued representation *2016 LV Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE). IEEE* pp 1353–8

[33] Takaki Y, Mitarai K, Negoro M, Fujii K and Kitagawa M 2021 Learning temporal data with a variational quantum recurrent neural network *Phys. Rev.* A **103** 052414

[34] Rivera-Ruiz M A, Mendez-Vazquez A and López-Romero J M 2022 Time series forecasting with quantum machine learning architectures *Mexican International Conference on Artificial Intelligence* (Springer) pp 66–82

[35] Xiang W, Li F, Wang J and Tang B 2018 Quantum weighted gated recurrent unit neural network and its application in performance degradation trend prediction of rotating machinery *Neurocomputing* **313** 85–95

[36] Song P and Zhang Z 2023 Research on multiple load short-term forecasting model of integrated energy distribution system based on mogrifier-quantum weighted MELSTM *Energies* **16** 3697

[37] Siemaszko M, Buraczewski A, Le Saux B and Stobińska M 2023 Rapid training of quantum recurrent neural networks *Quantum Machine Intelligence* **5** 31

[38] Padha A and Sahoo A 2022 A parametrized quantum LSTM model for continuous stress monitoring *2022 IX International Conference on Computing for Sustainable Global Development (INDIACom). IEEE* 261–6

[39] Hou X, Yang Y and Wang X 2022 Realization of long short-term memory networks on quantum circuits *2022 XIII Asian Control Conference (ASCC). IEEE* pp 2360–6

[40] Chen S Y-C, Yoo S and Fang Y-L L 2022 Quantum long short-term memory *ICASSP 2022-2022 IEEE International Conference on Acoustics* pp 8622–6

[41] 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** 130434130434–48

[42] Akter M S, Shahriar H and Bhuiya Z A 2022 Automated vulnerability detection in source code using quantum natural language processing *Inernational Conference on Ubiquitous Security* (Springer) pp 83–102

[43] Beaudoin C, Kundu S, Topaloglu R O and Ghosh S 2022 Quantum machine learning for material synthesis and hardware security *Proceedings of the XLI IEEE/ACM International Conference on Computer-Aided Design* pp 1–7

[44] Weigold M, Barzen J, Leymann F and Salm M 2021 Expanding data encoding patterns for quantum algorithms *2021 IEEE XVIII International Conference on Software Architecture Companion (ICSA-C). IEEE* pp 95–101

[45] Schuld M and Petruccione F 2021 *Machine Learning With Quantum Computers* (Springer)

[46] Holmes Z, Sharma K, Cerezo M and Coles P J 2022 Connecting ansatz expressibility to gradient magnitudes and barren plateaus *PRX Quantum* **3** 010313

[47] 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

[48] Sahoo B B, Jha R, Singh A and Kumar D 2019 Long short-term memory (LSTM) recurrent neural network for low-flow hydrological time series forecasting *Acta Geophys.* **67** 1471–81

[49] Ghanbari R and Borna K 2021 Multivariate time-series prediction using LSTM neural networks *2021 XXVI International Computer Conference, Computer Society of Iran (CSICC). IEEE* pp 1–5

[50] Rokhsatyazdi E, Rahnamayan S, Amirinia H and Ahmed S 2020 Optimizing LSTM based network for forecasting stock market *2020 IEEE congress on evolutionary computation (CEC). IEEE* pp 1–7

[51] Lindsay J and Zand R 2023 A novel stochastic LSTM model inspired by quantum machine learning *2023 XXIV International Symposium on Quality Electronic Design (ISQED). IEEE* pp 1–8

[52] Chehimi M, Chen S Y-C, Saad W and Yoo S 2024 Federated quantum long short-term memory (fedqlstm) *Quantum Machine Intelligence* **6** 1–13

[53] You X and Wu X 2021 Exponentially many local minima in quantum neural networks *International Conference on Machine Learning. PMLR* pp 12144–55

[54] Jones T and Gacon J 2020 Efficient calculation of gradients in classical simulations of variational quantum algorithms arXiv:2009.02823

[55] Jiménez-Guarneros M, Gómez-Gil P, Fonseca-Delgado R, Ramírez-Cortés M and Alarcón-Aquino V 2017 Long-term prediction of a sine function using a LSTM neural network *Nature-Inspired Design of Hybrid Intelligent Systems* (Springer) pp 159–73

[56] Panahi A, Saeedi S and Arodz T 2019 word2ket: Space-efficient word embeddings inspired by quantum entanglement arXiv: 1911.04975

[57] Chen Y, Pan Y and Dong D 2021 Quantum language model with entanglement embedding for question answering *IEEE Transactions on Cybernetics* **53** 3467–78