Quantum Recurrent Neural Networks for Multivariate Time Series Prediction

José Daniel Viqueira Cao, Daniel Faílde Balea, Mariamo Mussa Juane, Andrés Gómez Tato, and David Mera Pérez

Galicia Supercomputing Center (CESGA), 15705 Santiago de Compostela, Spain
Computer Graphics and Data Engineering (COGRADE), Departamento de Electrónica e Computación, Universidade de Santiago de Compostela, 15782 Santiago de Compostela, Spain
Correspondence: josedaniel.viqueira@rai.usc.es

DOI: *https://doi.org/10.17979/spu.23.c57*

Abstract: Classical models for time series forecasting, despite their success, often face challenges related to training, generalization, energy consumption and interpretability. Unconventional computing paradigms, such as quantum computing, offer a promising avenue to address these limitations. Quantum Recurrent Neural Networks (QRNN) emerge as a powerful approach for multivariate time series prediction. Due to some features, the QRNN model we study is not supported in many quantum computers. We develop a specific-purpose emulator to address this barrier, and then we benchmark the model against datasets of varying complexities, involving realistic features such as noisy outputs. The results show significant performance compared to other well-known classical approaches for time series prediction.

1 Introduction

The analysis of multivariate time series in real scenarios usually involves sophisticated algorithms that process high-dimensional variables evolving in time, wherein temporal correlations are not easy to find. In classical computation, Machine Learning is a consolidated approach for prediction and anomaly detection tasks. Recurrent Neural Networks (RNNs) are a powerful tool for modeling sequential data, although several variations over the first model have been shown to improve their performance, like the well-known *Long Short-Term Memory* (LSTM) or the *Gated Recurrent Unit* (GRU) cells. Transformers are popular alternative that but there is not enough work when using this algorithm for numerical multivariate time series.

Despite their success, RNN models cannot store information from early inputs in long time series. The LSTM cell was born to address this problem. However, the temporal correlations between different variables in multivariate time series and the complex patterns are still difficult to model.

Quantum Machine Learning (QML) leverages the power of Quantum Computing to produce complex patterns that are not likely to be efficiently computed in a classical computer Biamonte et al. (2017). In the current *Noisy Intermediate-Scale Quantum* (NISQ) era of quantum computers, there is a need for algorithms requiring shallow quantum circuits. Then, an important part of the research in quantum algorithms has been focusing on the Variational Quantum Algorithms (VQAs) Cerezo et al. (2021). A VQA relies on hybrid classical-quantum routines to classically minimise a cost function computed with a back-end Parameterised Quantum Circuit (PQC).

A VQA can replace a Neural Network by feeding a quantum circuit with our dataset and then tuning the circuit parameters to minimise a cost function in a supervised machine learning setting Mitarai et al. (2018). The QRNN model seeks to enhance the use of Quantum Computing

by a PQC that processes the time series, and the only classical part is data pre-processing and post-processing, as well as the optimisation of circuit parameters Viqueira et al. (2025).

The QRNN model should provide several advantages compared to classical neural networks. The encoding of classical data into quantum states allows an exponential growing of the number of basis functions on inputs as the number of qubits increases Mitarai et al. (2018). This feature generates complex patterns that are essential in real world scenarios. A classical computer would require exponential resources, which implies higher energy consumption, assuming the consumption is proportional to the number of qubits or bits. Besides that, the QRNN circuit involves intermediate measurements, becoming an open quantum system. This type of system has been shown to be a resource for quantum reservoir computing, a computing paradigm closely related to our model and commonly used for chaotic time series prediction Sannia et al. (2024). Finally, despite the counterintuitive nature of quantum physics, the open quantum system has measurable magnitudes, such as the entropy, and follows particular dynamics. These features would allow us to better understand how well and why (why not) the model works for a given problem.

In this work, we aim to show the usefulness of the QRNN and its performance in several datasets in comparison with similar classical approaches. The results shown here are intended to be a solid motivation for further research on the QRNN model, which needs to be further assessed in terms of memory and non-linear capacity for time series prediction, and its robustness against quantum hardware noise.

2 A Variational Quantum Algorithm as a recurrent model

The QRNN model is inspired by the classical RNN model. Since a quantum circuit diagram represents the series of operations applied over several qubits (vertical direction) during the time (horizontal direction), the QRNN circuit can be interpreted as the unrolled representation through time of the QRNN.

2.1 The classical Recurrent Neural Network

An input multivariate time series is a time-ordered dataset, $\{\boldsymbol{x}_{(t)}, t \in \{0, \ldots, T\}\}$, where each item is a vector of n_v components (variables). The output series is $\{y_{(t)}, t \in \{0, \ldots T\}\}$, which is the univariate target series in the supervised learning task.

Unlike feedforward neural networks in which inputs are processed independently, the output of a RNN at time t depends on the inputs from the previous time steps since it preserves past information with a form of memory. For the sake of simplicity, let's think in a box that continuously receives and returns data. Every time step, the box is supplied with an input, $\boldsymbol{x}_{(t)}$, and a hidden state, $\boldsymbol{h}_{(t-1)}$. It then returns two objects: an output $\overline{y}_{(t)}$ and a new *hidden state* $\boldsymbol{h}_{(t)}$ which is re-introduced in the box at the next time step. The model is represented in Fig. 1. Both $\overline{y}_{(t)}$ and $\boldsymbol{h}_{(t)}$ are functions that depend on $\boldsymbol{x}_{(t)}$ and $\boldsymbol{h}_{(t-1)}$,

$$\begin{cases} \overline{\boldsymbol{y}}_{(t)} & = \mathcal{Y}\left(\boldsymbol{x}_{(t)}, \boldsymbol{h}_{(t-1)}\right) \\ \boldsymbol{h}_{(t)} & = \mathcal{S}\left(\boldsymbol{x}_{(t)}, \boldsymbol{h}_{(t-1)}\right). \end{cases} \tag{48.1}$$

The functions $\mathcal{Y}$ and $\mathcal{S}$ are expensive to compute. They consist of matrix calculations that involve the tunable parameters (weights), structure and connections between the layers of neurons.

2.2 The QRNN circuit

The QRNN has PQCs implicitly performing matrix calculus. Like in the proposed classical model, wherein a network returns output data every time step, we measure the quantum circuit

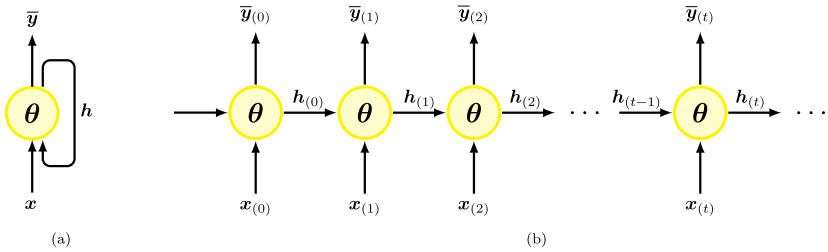

Figure 1: Classical Recurrent Neural Network representations. (a) RNN scheme. (b) RNN scheme unrolled through time.

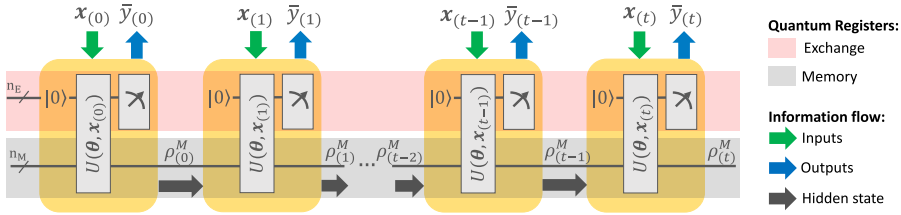

Figure 2: General form of the QRNN circuit. Arrows show the information flow.

to get the data in our classical devices. Quantum Networks also contain a *hidden-state*, now represented by ρ.

The circuit consists of two quantum registers: an *exchange register* (E) with n_E qubits, and a *memory register* (M) with n_M qubits. The former is used to exchange information between the quantum and the classical interface, by applying data encoding and measurement operations, while the latter is never measured. The state of register M after each measurement is the hidden state and it contains information from both registers E and M, provided that E and M are entangled before measurement. We say that E and M are entangled if their states cannot be described individually. If entangled, both parts have correlated information. The total number of qubits is $n \equiv n_E + n_M$.

We then have a circuit *block* for every time step: an estimation of an output from the input at time t. The instructions are (see Fig. 2):

1. Initialise register M to the $|0\rangle$ state.
2. Reset register E. All register E qubits start at $|0\rangle$ every time step.
3. Apply a parameterised unitary operator $U\left(\boldsymbol{x}_{(t)}, \boldsymbol{\theta}\right)$ that evolves the state of the circuit and entangles both registers E and M. $\boldsymbol{\theta}$ is the set of tunable parameters (weights) of the network.
4. Measure qubits from register E.
5. Repeat steps (2), (3), and (4) from $t = 1$ to $t = T$.

$\boldsymbol{\theta}$ is always the same for the different time steps, following the classical approach (see Fig. 1).

2.3 Quantum circuit emulation and gradients computation

We developed a specific-purpose emulator for QRNN in Python. The code can be found at *https://doi.org/10.5281/zenodo.12755194*, along with the detailed results of the benchmarks shown in the manuscript.

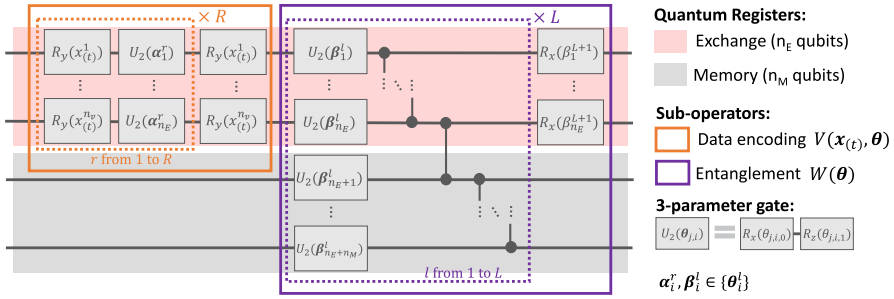

Figure 3: The QRNN ansatz $U(\boldsymbol{x}_{(t)}, \boldsymbol{\theta})$ consists of two parts. The first one is the data encoding, and gates inside the orange box are repeated with different parameters, that are a subset of trainable parameters, $\boldsymbol{\alpha}_i^r \in \{\boldsymbol{\theta}\}$. We use one qubit per input variable. The second part is for evolution and entanglement, where the blue box is repeated L times (layers). Each layer is a column of $R_z R_x$ rotations parameterised by a pair of parameters, $\boldsymbol{\beta}_i^l \in \{\boldsymbol{\theta}\}$, and a ladder of CZ gates. A final column of R_x gates is applied over register E before measurement.

	QRNN					cRNN			LSTM			GRU		
Case	n_E	n_M	L	R	N_θ	s_{in}	s_h	N_θ	s_{in}	s_h	N_θ	s_{in}	s_h	N_θ
(a)	2	2	4	1	39	1	4	33	1	2	43	1	2	33
(b)	2	3	5	3	65	2	6	67	2	2	51	2	3	67
(c)	1	2	5	3	38	1	4	33	1	2	43	1	2	33

Table 1: Quantum circuit and neural network configuration for each case analysed. In case (a), we reintroduce data onto the second qubit to improve the encoding expressivity. Classical neural networks: input size s_{in}, hidden size s_h and the corresponding number of weights N_θ. The output size s_{out} is consistently set to 1, and the number of layers L is fixed at 1, i.e., single-layered stacked neural networks.

The mathematical derivation for computing the evolution of the QRNN state through time and obtaining the output values uses the density matrix description of quantum states as described in Viqueira et al. (2025). With our method, we reduce the theoretical computational cost of the emulation in a factor of $(2^{n_E})^2$.

Due to the probabilistic nature of quantum measurements, the quantum circuit outputs are estimated after N_{shots} repetitions (shots) of identical quantum circuits, procedure known as sampling. This allows us to estimate probabilities and expectation values. The precision of the estimations is limited by the by the *sampling noise* which is gaussian noise with standard deviation bounded by $1/\sqrt{N_{shots}}$. Emulation in conventional computers allows us to compute probabilities and expectation values from a quantum circuit measurement with a precision only limited by the computer's numerical precision. However, when running the circuit in a quantum computer, we need to do sampling.

As forward differences methods for numerical gradient computation in machine learning require high numerical precision, analytical gradients are needed when sampling noise is present. The Parameter Shift Rule is a method for computing analytical derivatives of quantum circuit outputs Schuld et al. (2019). In QRNN we require $2N_\theta T$ circuit evaluations, being N_θ the number of trainable parameters. A circuit evaluation is the estimation of outputs from a circuit with a given set of input data and parameters encoded. A circuit evaluation in a quantum computer requires N_{shots} shots for sampling.

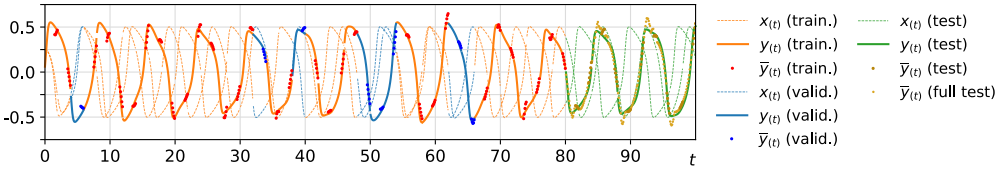

Figure 4: Result of the learning task with the QRNN model for case (b).

3 Results

For experiments (see Viqueira et al. (2025) for details), we define a quantum unitary operator for encoding input data and evolving the state of the circuit, so that we can implement non-linear functions on inputs and memory, as in Eq. 48.1. The operator $U(\boldsymbol{x}_{(t)}, \boldsymbol{\theta})$, used to test the performance of the QRNN model, is represented in Fig 3. The encoding part $V(\boldsymbol{x}_{(t)}, \boldsymbol{\theta})$ acts only over register E qubits and repeats the encoding for each input value $R+1$ times. The evolution and entanglement part $W(\boldsymbol{\theta})$ does not vary during the circuit since $\boldsymbol{\theta}$ is fixed inside a circuit evaluation. L entangling layers are added.

We show the tests on three normalised datasets: (a) a non-linear damping signal with a sinusoidal perturbation, a set (b) consisting of two non-linear damping signals as input and a linear combination of them as output, and (c) the Santa Fe laser series Hübner et al. (1989). The target is a series with temporal delays with respect to the inputs. For (c), we test the QRNN with two delays, $t_d = \{1, 10\}$. t_d equals the number of time steps the output is forward with respect to the input. To compare our model with existing algorithms, we have repeated the training with the classical RNN, LSTM and GRU models with similar number of weights.

The QRNN is trained by optimising the set of parameters $\boldsymbol{\theta}$, using the Adam algorithm. The series are divided into windows of $T = 20$ points and the task is predicting 5 points of the output variable. The windows are divided into three sets: 20 % for testing, 16 % for validation, and 64 % for training. The cost function for training is the Mean Squared Error (MSE) between the output $\overline{y}$ and the target series y of each training window (sample). The prediction for every time step is $\overline{y}_{(t)} = \langle Z^{\otimes n_E} \rangle_{(t)} + b$, where Z is the σ_z Pauli matrix and b is a trainable bias.

The network hyperparameters used for each case are indicated in Table 1. We systematically set the number of qubits and layers to have a network with sufficient complexity to make adequate predictions of non-trivial series. Indeed, in (a) we reintroduce data in the second qubit to improve the encoding expressivity.

We show dataset (b) as the representative multivariate example. In Fig. 4 we see the 2-variables input series divided by windows of 20 points and the corresponding 5-point predictions with the noiseless QRNN. The *full test* involves all the output points in the test region by shifting the windows 5 by 5 points. Fig. 5 shows the different training curves depending on whether we use numerical or analytical gradients and presence or absence of sampling noise. Without noise, numerical gradients achieve enough precision that the result is the same as for analytical gradients. In presence of noise, the optimisation is slower and achieves worse results when we decrease the number of shots.

Our ongoing research focusses on the effect of hardware noise on the optimisation. Different effects should be seen when this noise, which is not Gaussian, is added to our simulations. In fact, the effect is different when multiple Quantum Processing Units (QPU) are used to train the QRNN in a distributed framework. Distribution of gradient components in different QPUs induces biases with respect to the ideal derivatives, while distribution of shots does not directly induce bias but it can influence the optimisation speed and final loss.

Fig. 6 represents the comparison between classical models and the QRNN in the full test set. The performance of each model strongly depends on the dataset. The quantum model is always an alternative to other methods, because it can always achieve better results with a good selection of initialisation weights for training. In case (c), we see the improvement that

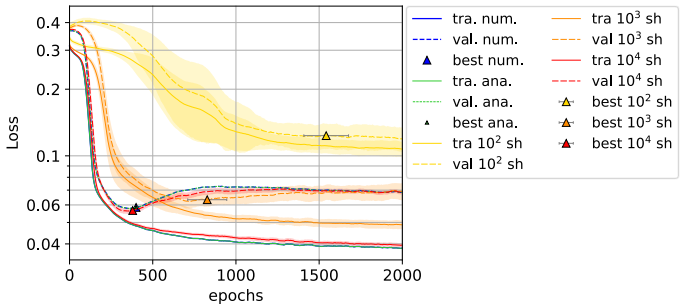

Figure 5: Training curves for dataset (c). The optimisations are those with lower final validation RMSE for numerical gradients. Optimisations with analytical gradients simulate sampling noise in every circuit evaluation with the number of shots specified in the legend. Lines of optimisations with sampling noise represent the mean of 8 different realisations and shaded regions represent the interval $\pm\sigma$ (standard deviation) around the mean. Triangles indicate the average epoch of minimum validation RMSE. The curves are plotted as a moving average of 50 points.

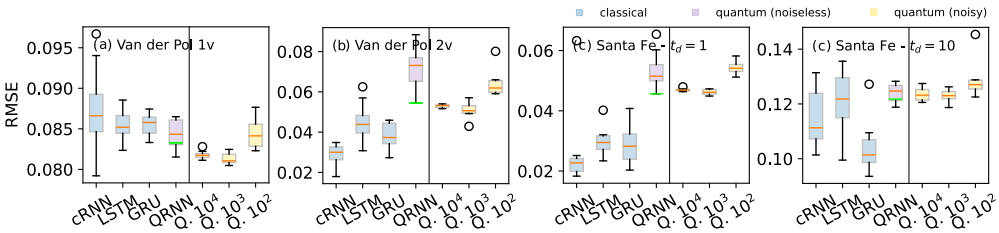

Figure 6: Comparison of full test RMSE results across different models used for making predictions on datasets. Box plots illustrate the distributions of RMSE for the classical models and the QRNN, based on 10 realizations with varied random parameter initializations. The box plots for QRNN ("Q. N_{shots}") with finite number of shots represent 8 realisations, all initialised with the same parameter set as the noiseless optimisation leading to the lowest validation RMSE. The green horizontal line indicates the RMSE value associated with the optimization that resulted in the lowest validation loss.

GRU offers when the delay is bigger, i.e., prediction is for a further future, due to its longer-term memory compared to vanilla RNN. Nonetheless, the quantum model offers comparable performance to cRNN and LSTM when the delay is bigger.

Continuing our line of research, we are looking for problem-agnostic metrics for the memory capacity. To this end, NARMA and Short Term Memory tasks, commonly used for assessing quantum reservoir computing settings, are useful. The correlation between memory capacity and physical magnitudes, such as the entropy, will also help for a better interpretability of the studied structure. Finally, we are finding new quantum operator structures for improving the encoding of multiple variables and enabling the finding of temporal correlations between them.

4 Discussion

Our work establishes the QRNN as a competitive model in the toolkit of machine learning algorithms for multivariate time series prediction, because it shows similar accuracies, prediction and generalisation capabilities. Furthermore, a strong feature of QRNN is that it requires a low number of qubits. Sampling noise, which is avoidable in emulation but not in actual quantum computers, has shown to be not critical because analytical gradients with noise allow parameter optimisations ending in close accuracies and number of epochs comparable to noiseless

optimisations. Our current line of research extends this work towards a more realistic scenario, where hardware noise is present.

Hyperparameter search to enhance the power of the neural network, adaptability to the quantum hardware limitations, research on circuit structures that capture correlations behind complex datasets and the adaptability of different optimisation techniques are starting points towards an extended use of Quantum Recurrent Neural Networks to multivariate time series prediction in real life.

Acknowledgments

We thank the CESGA Quantum Computing researchers for their feedback and the stimulating intellectual environment they provide. This work was supported by Axencia Galega de Innovación through the Grant Agreement "Despregamento dunha infraestructura baseada en tecnoloxías cuánticas da información que permita impulsar a I+D+I en Galicia" within the program FEDER Galicia 2014-2020. This work was partially supported by the Galician Government under Grant ED431B 2024/44. A. Gómez, D. Faílde and M. M. Juane were supported by MICIN through the European Union NextGenerationEU recovery plan (PRTR-C17.I1), and by the Galician Regional Government through the "Planes Complementarios de I+D+I con las Comunidades Autónomas" in Quantum Communication. J. D. Viqueira was supported by Axencia Galega de Innovación (Xunta de Galicia) through the "Programa de axudas á etapa predoutoral". Simulations on this work were performed using Galicia Supercomputing Center (CESGA) FinisTerrae III supercomputer with financing from the Programa Operativo Plurirregional de España 2014-2020 of ERDF, ICTS-2019-02-CESGA-3, and the Qmio quantum infrastructure, with financing from the European Union, through the Programa Operativo Galicia 2014-2020 of ERDF_REACT EU, as part of the European Union's response to the COVID-19 pandemic.

Bibliography

J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd. Quantum machine learning. *Nature*, 549(7671):195–202, 2017.

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

U. Hübner, N. B. Abraham, and C. O. Weiss. Dimensions and entropies of chaotic intensity pulsations in a single-mode far-infrared NH 3 laser. *Physical Review A*, 40(11):6354–6365, 1989.

K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii. Quantum circuit learning. *Physical Review A*, 98(3), 2018.

A. Sannia, R. Martínez-Peña, M. C. Soriano, G. L. Giorgi, and R. Zambrini. Dissipation as a resource for Quantum Reservoir Computing. *Quantum*, 8:1291, 2024.

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

J. D. Viqueira, D. Faílde, M. M. Juane, A. Gómez, and D. Mera. Density matrix emulation of quantum recurrent neural networks for multivariate time series prediction. *Machine Learning: Science and Technology*, 6(1):015023, 2025.