EVOLVING HYBRID QUANTUM-CLASSICAL GRU ARCHITECTURES FOR MULTIVARIATE TIME SERIES

Francesca De Falco, Leonardo Lavagna, Andrea Ceschini, Antonello Rosato, Massimo Panella

Department of Information Engineering, Electronics and Telecommunications
University of Rome “La Sapienza”, Via Eudossiana 18, 00184 Rome, Italy
{francesca.defalco,andrea.ceschini,massimo.panella}@uniroma1.it

2024 IEEE 34th International Workshop on Machine Learning for Signal Processing (MLSP) | 979-8-3503-7225-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/MLSP58920.2024.10734792

ABSTRACT

Quantum Recurrent Neural Networks are gaining attention for their generalization capability in time series analysis. However, their performance are hindered by lengthy training times and non-scalability. This paper proposes a novel application for hybrid Quantum Gated Recurrent Units (QGRUs) focused on multivariate time-series forecasting. Our study demonstrates that these architectures outperform classical benchmarks. Through extensive innovative experiments and simulations, our results showcase the versatility and superiority of the hybrid approach, extending the capabilities of QGRUs especially in multidimensional data applications. In addition, the architecture at the basis of the QGRU has 25% fewer quantum parameters than existing Quantum Long Short-Term Memory models, and it is about 25% faster during training and inference stages, leading to feasible implementations on both simulated and real quantum hardware.

***Index Terms*—** Quantum Machine Learning, Quantum Computing, Quantum Gated Recurrent Units, Multivariate Time Series

1. INTRODUCTION

In the past few years, the automation of complex tasks has undergone a significant transformation, as Machine Learning (ML) and Deep Learning (DL) have become essential instruments for addressing hard challenges. These technologies have demonstrated strong capabilities in diverse domains, from computer vision [1] to anomaly detection [2] and beyond. In particular, Recurrent Neural Networks (RNNs) such as Long Short-Term Memory (LSTM) networks [3] have been widely adopted for their ability to model temporal dependencies in sequential data without succumbing to the vanishing gradient problem [4]. Despite their success, a substantial computational burden arises from the intricate gate-based architecture [5]. To mitigate this problem, Gated Recurrent Unit (GRU) networks [6] have been introduced as a more efficient alternative. With a simplified architecture and faster convergence, GRUs, composed solely of a reset gate r_t and an update gate z_t as shown in Fig. 1, demonstrate comparable performance to LSTMs [7].

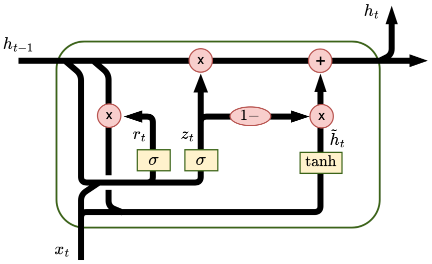

Fig. 1. Internal structure of a classical GRU cell.

Simultaneously, the field of DL is witnessing the emergence of an innovative trend with the introduction of Quantum Neural Networks (QNNs) [8–10], that are positioned to surpass their classical counterparts, utilizing the principles of quantum mechanics. QNNs exhibit efficiency in handling large-scale heterogeneous data and conducting effective high-dimensional processing. However, despite constant quantum hardware improvement, serious limitations still hamper the full-scale capabilities of Quantum Machine Learning (QML) [11]. In this sense, anticipating the realization of fault-tolerant quantum computers, researchers have made significant strides in developing quantum algorithms tailored for execution on Noisy Intermediate-Scale Quantum (NISQ) devices [12].

Inspired by [10, 13, 14], in this work we propose a novel application of the hybrid quantum-classical GRU, denoted as Quantum Gated Recurrent Unit (QGRU), where the hybrid architecture is employed to solve challenging multidimensional problems, seeking efficiency advantages. In particular, we focused on multivariate time-series analysis and prediction; to

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.

the best of our knowledge, this is the first instance where a hybrid QGRU is used for such a task. We carefully implemented the architecture showing that there is potential to outperform benchmarks such as the classical GRU, the classical LSTM, the classical Bi-LSTM, and the Quantum Long-Short Term Memory (QLSTM) [15]. In this sense, we have shown through experiments and simulations that the QGRU outperforms the benchmarks on different multidimensional datasets, extending the applications' range, and improving upon state-of-the-art models. In addition, the QGRU has 25% fewer quantum parameters with respect to the QLSTM, yielding results to be more easily feasible for current quantum devices.

2. VARIATIONAL LEARNING AND HYBRID QUANTUM-CLASSICAL ARCHITECTURES

Variational Quantum Circuits (VQCs) are the most promising solution for executing QML algorithms on NISQ devices. They incorporate quantum computations into classical Neural Networks (NNs), thus defining hybrid quantum-classical architectures, and offer a versatile framework for solving various problems across multiple domains [16].

2.1. Variational Quantum Algorithms

Variational Quantum Algorithms (VQAs) describe the theoretical framework in which hybrid quantum-classical architectures are developed, and their implementation is based on VQCs. The fundamental elements and basic principles at the core of VQAs are summarized below, and sketched in Fig. 2.

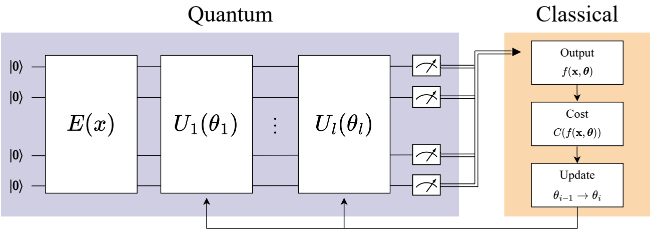
Quantum
Classical
|0⟩
|0⟩
|0⟩
|0⟩
$E(x)$
$U_1(\theta_1)$
$U_l(\theta_l)$
Output
$f(\mathbf{x}, \boldsymbol{\theta})$
Cost
$C(f(\mathbf{x}, \boldsymbol{\theta}))$
Update
$\theta_{i-1} \to \theta_i$

Fig. 2. Scheme of a VQC for supervised learning.

2.1.1. Data Encoding

Classical data points $\{x_1, \ldots, x_n\} \subset \mathbb{R}^n$ are encoded into quantum states as elements of a finite-dimensional Hilbert space $\mathbb{H} \subset (\mathbb{C}^2)^{\otimes n}$ via a quantum feature map of the form $x \to |\phi(x)\rangle := E(x)\,|\psi_0\rangle$, where $E = E(x)$ is a unitary operator acting on an initial state $|\psi_0\rangle$, usually $|\psi_0\rangle = |0\rangle^{\otimes n}$.

A suitable and efficient representation of classical input data into a quantum system strongly affects the performance of the underlying algorithm. To obtain a quantum advantage, the choice of the feature map must take into account the structure of the dataset, and possible relationships between data points. For this purpose, angle encoding with parametrized quantum gates is commonly employed for continuous input variables, where each feature is associated to a qubit by means of one or more data-dependent rotation gates. In this case, n qubits allow encoding n continuous variables.

2.1.2. Ansatz

An ansatz $U(\boldsymbol{\theta})$ of $\boldsymbol{\theta}$-parametrized unitaries with randomly initialized parameters and fixed entangling gates is applied to the encoded data $|\phi(x)\rangle$ repeatedly. The ansatz is usually represented as the product of l sequential unitary transformations $U(\boldsymbol{\theta}) = U_l(\theta_l) \cdots U_2(\theta_2)U_1(\theta_1)$, where $\boldsymbol{\theta}$ is an array of real parameters, and $U_j(\theta_j) = e^{-i\theta_j H_j}$ is the unitary operator associated with the j-th observable H_j of interest. Typically, H_j is represented by one of the Pauli matrices $\{\boldsymbol{\sigma}_x, \boldsymbol{\sigma}_y, \boldsymbol{\sigma}_z\}$.

2.1.3. Measurement

A measurement process yielding a prediction as the expectation value of a given observable M, of the form

$$f(x, \boldsymbol{\theta}) = \langle\phi(x)|\, U(\boldsymbol{\theta})^\dagger M U(\boldsymbol{\theta})\, |\phi(x)\rangle \ . \qquad (1)$$

After the execution of a VQC, a cost function $C(f(x, \boldsymbol{\theta}))$ is evaluated, and the parameters $\boldsymbol{\theta}$ are updated through a query to a classical device. The whole cycle between quantum and classical hardware is run iteratively to search for better parameters at every step, in a fashion similar to the training of neural networks.

2.2. Hybrid Quantum-Classical Neural Networks

Given a classical NN architecture, integrating a VQC can enhance accuracy, reduce training parameters, or significantly shorten the training phase. As a guiding example consider the GRU (Fig. 1) with n inputs and k outputs related to a given time series $X := (x_t)_{t \in I \subset \mathbb{R}}$. The basic GRU cell can be enhanced using a VQC, where the dynamics governing the classical time-evolution of the cell are given by the learnable parameters

$$\begin{cases} z_t = \sigma(\mathbf{a}_z x_t + \mathbf{b}_z h_{t-1} + c_z) \\ r_t = \sigma(\mathbf{a}_r x_t + \mathbf{b}_r h_{t-1} + c_r) \\ \tilde{h}_t = \psi(\mathbf{a}_h x_t + \mathbf{b}_h (r_t \odot h_{t-1}) + c_h) \end{cases} \qquad (2)$$

where $\Omega = (\mathbf{a}_\omega, \mathbf{b}_\omega, c_\omega)_{\omega \in \{z,r,h\}} \in \mathbb{R}^{n \times k} \times \mathbb{R}^{n \times k} \times \mathbb{R}^k$, z_t and r_t are the update and reset vectors, respectively, σ is the logistic activation function, ψ is the hyperbolic tangent activation function, and $\tilde{h}_t$ is the candidate activation vector defining the output vector $h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t$ as a weighted combination of the Hadamard product between z_t and h_{t-1} and the Hadamard product between z_t and $\tilde{h}_t$.

The GRU dynamics can be modified by incorporating a quantum reset, update, and output stepas shown in Fig. 3. The main components of such a hybrid architecture are:

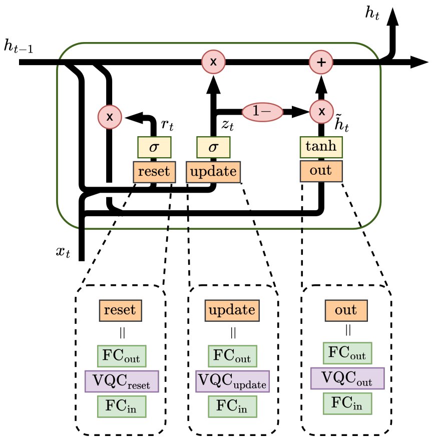

Fig. 3. Internal scheme of a QGRU cell. The reset gate, update, and output gates are substituted with three VQCs. Every VQC takes as inputs the results of the corresponding FC_{in}, and outputs the results to the corresponding FC_{out}.

1. The VQCs implementing the reset, update, and output steps (the VQC_{reset}, VQC_{update}, and VQC_{out} in Fig. 3);
2. The Fully Connected (FC) classical layers (i.e., FC_{in} and FC_{out} in Fig. 3) to manage the input-output stages for the VQCs, where qubit limitations and consistency relations in the interface between the classical architecture and the quantum device must be dealt with.

At this point, the parameters $\boldsymbol{\theta}$ associated with the VQC can be trained together with the parameters Ω of the "vanilla" GRU cell using standard well-known methods.

3. EXPERIMENTAL METHODOLOGY AND EVALUATION

Enhancing the GRU model with quantum layers highlights the importance to carefully choose both encoding strategy and VQC ansatz, as well as to implement and test the relevant models. To test the architectures, we utilized Google Colaboratory™ with standard resources and implemented the models in Python (v. 3.11) using standard libraries and modules, alongside PennyLane (v. 0.34.0) and PyTorch.

3.1. Models' implementation

For the variational part of the adopted VQCs, many alternatives are possible. We used angle encoding and the Basic Entangler layer from PennyLane as ansatz, given by an initial layer of parametrized rotation gates followed by CNOT gates applied to every pair of qubits. Both the encoding and the ansatz blocks rely on R_x rotations, as shown in Fig. 4. If necessary, the ansatz can be repeated in blocks of $l > 1$ successive layers, concluding with a final measurement process.

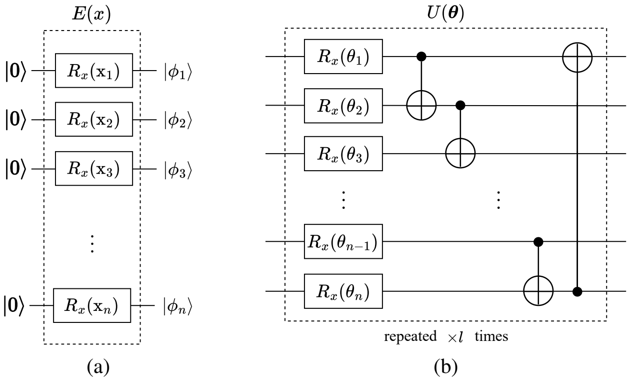

Fig. 4. (a) Angle encoding. (b) High-level structure of a Basic Entangler layer.

Table 1. Network Structures and Related Hyperparameters

Model	d_{hid}	qubits	l
QGRU	5	5	2
QLSTM	5	5	2
GRU	5	–	–
LSTM	5	–	–
Bi-LSTM	5	–	–

We restricted our attention to this latter encoding scheme and ansatz due to the nature of the addressed problems and the datasets considered. Concerning the choice of the ansatz, after an extensive grid search procedure, we heuristically found that the Basic Entangler layer yields the best performance in terms of Mean Squared Error (MSE) and Mean Absolute Error (MAE). As shown in Fig. 3, the proposed QGRU is also characterized by the use of identical FC inputs, as well as identical FC outputs.

The QGRU model is compared with an LSTM model, a GRU, and a Bi-LSTM model. Additionally, a comparison with QLSTM is proposed, implementing it in such a way that the number of parameters is comparable to those of QGRU (considering the input FCs and the output FCs of the various cells all equal). We evaluated every model alongside a Naive predictor, a straightforward yet widely employed benchmark for time series forecasting. The Naive predictor operates on the assumption that the upcoming value in the time series will mirror the most recent observed value, serving as a foundational comparison for assessing the efficacy of more complex models, enabling us to determine whether the proposed quantum-inspired architectures yield superior results compared to this simple heuristic approach.

The considered networks are summarized 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. In our implementation we fixed the hyperparameters as listed in Table 2. The window size is set

Table 2. Training Hyperparameters

Experiment	n_e	lr	t_{lr}	lr_d
TEMP3	490	0.01	50	0.9
ETTH1	310	0.01	50	0.9
ELIZABETH CITY OCT. 2013	500	0.01	150	0.8
SACRAMENTO OCT. 2013	480	0.01	150	0.8

to $w = 5$, chosen to strike a balance between running time and prediction capabilities of the tested architectures. The number of epochs n_e was tailored on the dataset used. The initial learning rate lr was then adjusted after a drop period t_{lr} of 50 epochs and multiplied by a common drop factor $lr_d = 0.9$ to enhance the training phase in the case of datasets TEMP3 and ETTH1. In the other two cases, we proved that to improve the training phase, it was optimal to change the initial learning rate after a drop period t_{lr} of 150 epochs and multiplied by a common drop factor $lr_d = 0.8$. The hyperparameters were found using a grid search approach, striking a balance between running time and MSE.

3.2. Complexity analysis

Let d_{hid} represent the networks' hidden dimension, n the number of qubits per quantum circuit, d_{in} the input feature count, and l the number of ansatz layers in each VQC. Each VQC layer comprises $n \cdot l$ parameters. The classical layer FC_{in} takes d_{conc} input features, where $d_{conc} = d_{hid} + d_{in}$ is the dimension of the concatenated vectors $[h_{t-1}, x_t]$, and produces an n-dimensional output vector. Similarly, FC_{out} receives n input features and yields a vector with d_{hid} features. Thus, FC_{in} and FC_{out} each contain $d_{conc} \cdot n + n$ and $n \cdot d_{hid} + d_{hid}$ parameters, respectively.

A QLSTM inspired from [15] is composed of 3 gates and 1 cell buffer, so it has four VQCs, one FC_{in}, and one FC_{out} trainable layers, yielding a total number of parameters for a QLSTM cell of $4(n \cdot l) + (d_{conc} \cdot n + n) + (d_{hid} \cdot n + d_{hid}) = n(4l + 2d_{hid} + d_{in} + 1) + d_{hid}$. The number of quantum parameters is $4(n \cdot l)$. Similarly, a QGRU cell has 3 VQCs, 1 FC_{in}, and 1 FC_{out} trainable layers, yielding a number of parameters in a QGRU cell of $n \cdot (3l + 2d_{hid} + d_{in} + 1) + d_{hid}$, where only $3(n \cdot l)$ are quantum parameters. Consequently, QGRU uses 25% fewer quantum parameters than QLSTM, leading to faster computation.

3.3. Datasets and metrics

To test our architecture, we focused on multivariate time series, particularly on the Electricity Transformer Dataset (ETDataset) [17], the Elizabeth City State University dataset, the Sacramento Municipal Utility District dataset, and the New York City's temperatures dataset. The ETDataset comprises observations on multiple Electricity Transformers. Specifically, we focused on the ETT-SMALL subset of the ETDataset, particularly on the ETTH1 data available in the public repository `github.com/zhouhaoyi/ETDataset/tree/main`. The Elizabeth City State University dataset and the Sacramento Municipal Utility District dataset were both acquired from the Measurement and Instrumentation Data Center, which provides irradiance and meteorological data from different stations. The data are available at the following link: `midcdmz.nrel.gov`.

For the Elizabeth City State University dataset, we collected data spanning from October 1, 2013, to October 31, 2013, at the geographic coordinates of 36.282 degrees North latitude and 76.216 degrees West longitude. We decided to work specifically with a multivariate dataset consisting of three variables: the total solar radiance $[W/m^2]$, which served as the target variable for forecasting, the ambient air temperature [°C], and the relative humidity [%].

As for the Sacramento Municipal Utility District dataset, the data collected here are also from October 1, 2013, to October 31, 2013, at the geographic coordinates of 38.54586 degrees North latitude and 121.24029 degrees West longitude, and consists of the same three variables.

New York City's temperature data were acquired from the Prediction of Worldwide Energy Resource Project at NASA's Langley Research Center, publicly available at the following link: `power.larc.nasa.gov/data-access-viewer`. Specifically, we collected data in the TEMP3 dataset spanning from January 1, 2018, to December 31, 2022, at the geographic coordinates of 40.6745 degrees latitude and -73.9519 degrees longitude. This dataset comprises four variables: Earth Skin temperature [°C], which served as the target variable for forecasting, Wind Speed at 2 meters above ground level [m/s], Specific Humidity at 2 meters above ground level [g/kg], and Surface Pressure [kPa]. All the above datasets were split into 80% training and 20% test, then scaled with MinMaxScaler fitted on the training set in the range $[-1, 1]$.

The MSE and MAE are used as the main metrics to assess the performance across the different architectures on the selected datasets.

4. EXPERIMENTAL RESULTS

The main results we obtained on the TEMP3, ETTH1, SACRAMENTO, and ELIZABETH datasets are reported in Table 3. In particular, for the SACRAMENTO and ELIZABETH datasets the QGRU achieves the best performance even when compared to richer classical architectures such as the Bi-LSTM. In the case of the ELIZABETH dataset, the QGRU achieves an improvement of at least 8% in MSE compared to other classical architectures, while in the case of the SACRAMENTO dataset, the improvement is at least 7% in MSE.

Furthermore, there is also the advantage of a significant reduction in parameters: about 28% fewer than the case of the classical GRU, 46% fewer than the classical LSTM, and 73% fewer than the Bi-LSTM.

In the TEMP3 dataset, the architecture that achieves the best results in MSE is now the Bi-LSTM, but the QGRU still manages to obtain similar results in MSE and better results in MAE while maintaining an extremely lower number of parameters. Finally, on the ETTH1 dataset, it is again the QGRU that surpasses the performance of any other architecture. These results show consistency in the accuracy improvements yielded by our hybrid model on complex forecasting tasks arising from multivariate time-series analysis.

Illustrative results of such a task are reported in Fig. 5, Fig. 7, and Fig. 8 where the QGRU is trained on datasets TEMP3, ELIZABETH, and SACRAMENTO, respectively. One can observe how the quantum model accurately predicts the time series, capturing the fluctuating and spiking behavior correctly. Similarly, in Fig. 6, despite the difficulty of the task to model as the ETTH1 dataset does not exhibit the periodicity that characterizes the other datasets tested, the quantum model performs excellently, as the predictions closely match the expected theoretical results.

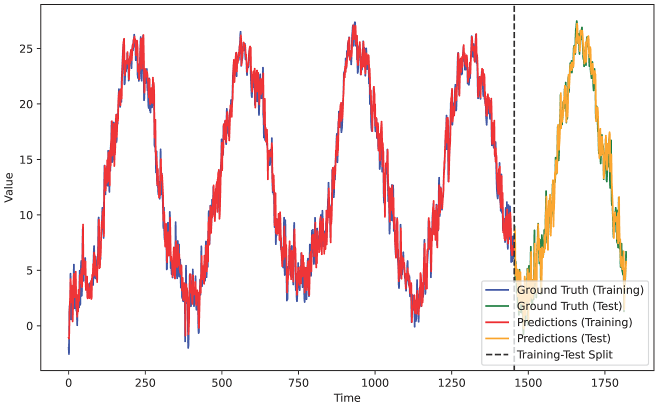

Fig. 5. QGRU prediction on Training and Test set compared to the related ground truth of the TEMP3 dataset. The black dashed line represents the 80%-20% Training-Test data split.

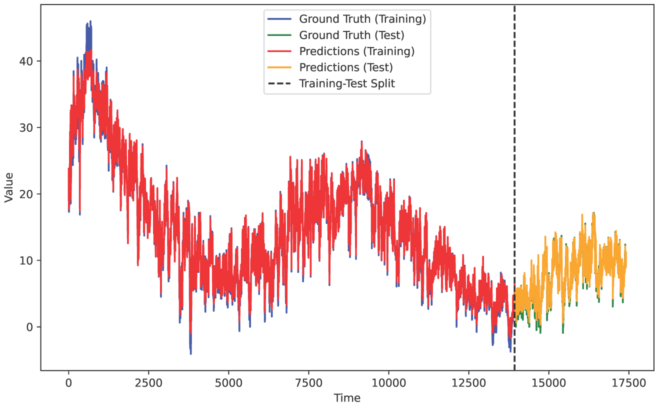

Fig. 6. QGRU prediction on Training and Test set compared to the related ground truth of the ETTH1 dataset.

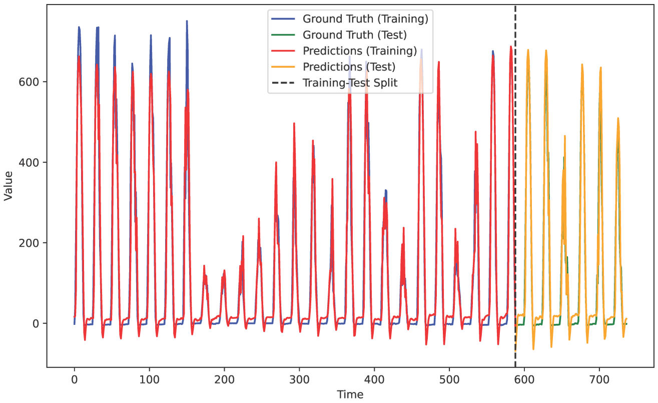

Fig. 7. QGRU prediction on Training and Test set compared to the related ground truth of the ELIZABETH dataset.

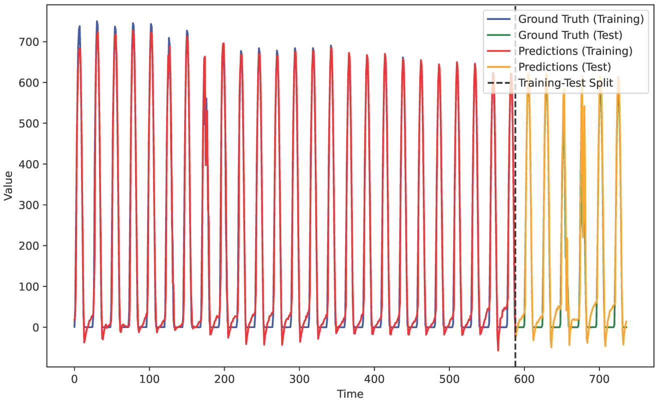

Fig. 8. QGRU prediction on Training and Test set compared to the related ground truth of the SACRAMENTO dataset.

5. CONCLUSIONS

In conclusion, our work addresses the challenges faced by quantum recurrent neural networks in time series analysis, especially their prolonged training times and non-scalable architectures. Through the application of the QGRU, a hybrid quantum-classical architecture inspired by recent advancements, we have achieved substantial improvements in performance on challenging tasks. Focused on the novel application of multivariate time-series analysis and prediction, the QGRU outperforms both classical benchmarks and related quantum architectures. Our extensive experiments and simulations datasets demonstrate the versatility and superiority of the proposed model, setting a new benchmark in the field of multidimensional data applications. In addition, our model has about 25% less parameters than existing quantum recurrent neural networks, leading to a more efficient and streamlined architecture for NISQ devices. Future research may involve the use of an actual quantum hardware for validation and computational times' assessment.

Table 3. Performance of Tested Architectures on the Considered Datasets

Dataset	Metrics	Naive	Bi-LSTM	GRU	LSTM	QGRU	QLSTM
TEMP3	MSE	1.608	1.129 ± 0.064	1.150 ± 0.083	1.166 ± 0.066	1.140 ± 0.027	1.151 ± 0.025
	MAE	0.995	0.821 ± 0.033	0.0829 ± 0.039	0.839 ± 0.035	0.806 ± 0.014	0.809 ± 0.015
	Parameters		451	171	226	116	126
ETTH1	MSE	0.428	0.588 ± 0.074	0.482 ± 0.045	0.549 ± 0.093	0.467 ± 0.042	0.538 ± 0.125
	MAE	0.448	0.644 ± 0.025	0.497 ± 0.029	0.543 ± 0.068	0.481 ± 0.026	0.531 ± 0.080
	Parameters		571	216	286	131	141
ELIZABETH	MSE	6847.642	3186.971 ± 198.011	3064.911 ± 290.041	3097.254 ± 175.156	2809.113 ± 205.450	2968.172 ± 234.376
	MAE	48.554	35.336 ± 1.819	35.264 ± 2.782	35.675 ± 2.813	34.018 ± 2.1386	34.4615 ± 1.745
	Parameters		411	156	206	111	121
SACRAMENTO	MSE	6812.106	3432.431 ± 143.901	3634.355 ± 278.844	3466.509 ± 391.959	3190.656 ± 326.413	3341.125 ± 278.619
	MAE	49.025	34.998 ± 2.085	35.694 ± 1.922	36.911 ± 1.471	35.728 ± 1.101	35.502 ± 1.605
	Parameters		411	156	206	111	121

6. REFERENCES

[1] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, “Deep Learning for Computer Vision: A Brief Review,” *Computational Intelligence and Neuroscience*, vol. 2018, p. 7068349, Feb. 2018.

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

[3] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[4] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks,” in *Int. Conf. on Machine Learning*. Pmlr, 2013, pp. 1310–1318.

[5] Y. Yu, X. Si, C. Hu, and J. Zhang, “A review of recurrent neural networks: LSTM cells and network architectures,” *Neural Computation*, vol. 31, no. 7, pp. 1235–1270, 2019.

[6] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, “On the Properties of Neural Machine Translation: Encoder-Decoder Approaches,” Oct. 2014.

[7] S. Yang, X. Yu, and Y. Zhou, “LSTM and GRU neural network performance comparison study: Taking yelp review dataset as an example,” in *2020 International workshop on electronic communication and artificial intelligence (IWECAI)*. IEEE, 2020, pp. 98–101.

[8] F. Tacchino, C. Macchiavello, D. Gerace, and D. Bajoni, “An artificial neuron implemented on an actual quantum processor,” *npj Quantum Information*, vol. 5, no. 1, p. 26, 2019.

[9] B.-Q. Chen and X.-F. Niu, “A novel neural network based on quantum computing,” *Int. Journal of Theoretical Physics*, vol. 59, no. 7, pp. 2029–2043, 2020.

[10] A. Ceschini, A. Rosato, and M. Panella, “Hybrid quantum-classical recurrent neural networks for time series prediction,” in *2022 international joint conference on neural networks (IJCNN)*. IEEE, 2022, pp. 1–8.

[11] K. A. Tychola, T. Kalampokas, and G. A. Papakostas, “Quantum machine learning—an overview,” *Electronics*, vol. 12, no. 11, p. 2379, 2023.

[12] J. Preskill, “Quantum Computing in the NISQ era and beyond,” *Quantum*, vol. 2, p. 79, 2018.

[13] R. Di Sipio, J.-H. Huang, S. Y.-C. Chen, S. Mangini, and M. Worring, “The dawn of quantum natural language processing,” in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 8612–8616.

[14] A. Ceschini, A. Rosato, and M. Panella, “Design of an LSTM cell on a quantum hardware,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 3, pp. 1822–1826, 2022.

[15] S. Y.-C. Chen, S. Yoo, and Y.-L. L. Fang, “Quantum long short-term memory,” in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 8622–8626.

[16] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio *et al.*, “Variational quantum algorithms,” *Nature Reviews Physics*, vol. 3, no. 9, pp. 625–644, 2021.

[17] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, “Informer: Beyond efficient transformer for long sequence time-series forecasting,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 12, 2021, pp. 11 106–11 115.