Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Architecture-Dependent Utility of Variational Quantum Gates in Recurrent Weather Forecasting: A Controlled QGRU–QLSTM Evaluation

Abhishek Tiwari

abhishek@cdac.in

Centre for Development of Advanced Computing

Bhushan Kape

Centre for Development of Advanced Computing

Pankaj Tyagi

University of Delhi

Sachin Kumar

University of Delhi

Geeta Singh

Ministry of Electronics and Information Technology

Research Article

Keywords: quantum machine learning, variational quantum circuit, quantum recurrent neural network, QGRU, QLSTM, meteorological forecasting, architecture-matched control, parameter-matched benchmarking

Posted Date: August 14th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10524017/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Architecture-Dependent Utility of Variational Quantum Gates in Recurrent Weather Forecasting: A Controlled QGRU–QLSTM Evaluation

Abhishek Tiwari[1*†], Bhushan Kape[1*†], Pankaj Tyagi[2*], Sachin Kumar[2] and Geeta Singh[3]

[1*]Centre for Development of Advanced Computing (C–DAC), Noida, 201307, Uttar Pradesh, India .
[2]Cluster Innovation Centre, University of Delhi, Delhi, 110007, India .
[3]R&D in Information Technology, Ministry of Electronics & Information Technology, New Delhi, 110003, India .

*Corresponding author(s). E-mail(s): abhishek@cdac.in; kapebhushan3872@gmail.com; pankajtyagi@cic.du.ac.in;
Contributing authors: skumar@cic.du.ac.in; geeta.singh30@meity.gov.in;
†These authors contributed equally to this work.

Abstract

Hybrid quantum recurrent models are commonly evaluated against compact classical networks, making it difficult to determine whether observed gains arise from the variational quantum circuit (VQC) or from the surrounding recurrent architecture. This study presents an architecture and parameter-controlled evaluation of quantum gated recurrent unit (QGRU) and quantum long short-term memory (QLSTM) models for one-step-ahead meteorological forecasting. Each quantum model is compared with both a compact, approximately parameter-matched recurrent baseline and an exact classical gate-core surrogate that preserves the hidden width, recurrent equations, gate-specific pre-/post-projections, optimization protocol, and total trainable-parameter count. All models use 17 input features, 24-step histories, a chronological 80/10/10 train–validation–test split, training-only standardization, and ten matched random seeds. QGRU achieves an RMSE of $\mathbf{1.1045 \pm 0.0693}$, reducing test RMSE by 19.01% relative to the compact GRU and by 5.47% relative to its exact 773-parameter surrogate. It also attains a lower mean best validation MSE, reaches 95% of its achievable validation improvement after $\mathbf{5.5 \pm 0.5}$ epochs, and exhibits a smaller final train–validation gap. QLSTM reduces test RMSE by 24.80% relative to the compact LSTM, while

its exact 1,025-parameter surrogate achieves comparable predictive performance with slightly greater validation stability. Analysis of 1,949 epoch-level records across convergence, checkpoint quality, early stopping, generalization, and runtime demonstrates that VQC utility depends on the recurrent architecture in which it is embedded. All experiments are conducted using ideal noiseless statevector simulation.

Keywords: quantum machine learning, variational quantum circuit, quantum recurrent neural network, QGRU, QLSTM, meteorological forecasting, architecture-matched control, parameter-matched benchmarking

1 Introduction

Time-series forecasting is a central problem in meteorology, energy systems, finance, transportation, and environmental monitoring. Recurrent neural networks, particularly gated recurrent units (GRUs) and long short-term memory (LSTM) networks, remain widely used because their gated state-update mechanisms can represent nonlinear temporal dependencies while mitigating the vanishing-gradient problem. Hybrid quantum-classical recurrent models extend these architectures by replacing some classical gate transformations with variational quantum circuits (VQCs). The resulting QGRU and QLSTM models are motivated by the possibility that a small parametrized quantum feature map may provide a useful nonlinear inductive bias under a restricted parameter budget.[1, 2]

The central scientific opportunity is to determine when the quantum gate itself contributes beyond the surrounding hybrid architecture. A hybrid quantum recurrent cell contains classical input projections, bounded angle encoding, a VQC, classical output projections, and a recurrent state whose width may differ from a compact baseline. A controlled experiment must therefore separate two effects: the predictive benefit of the complete hybrid recurrent architecture and the additional benefit attributable specifically to the VQC gate core. This distinction enables an affirmative but properly localized conclusion about where quantum nonlinearities are useful.[6, 8]

This paper addresses that attribution problem through two complementary comparison regimes. The first is a *parameter-budget comparison*, in which QGRU and QLSTM are compared with compact vanilla GRU and LSTM models having nearly equal total trainable parameter counts. The second, and scientifically more decisive, is an *architecture-matched gate-core ablation*. In this regime, the complete recurrent architecture is held fixed including hidden width, number of gates, independent per-gate projections, optimizer, data split, model-selection procedure, and total trainable parameters and the VQC is replaced by a classical nonlinear module with exactly the same number of trainable gate-core parameters.

The study investigates the following research questions:

1. **RQ1:** Do QGRU and QLSTM outperform compact, approximately parameter-matched vanilla GRU and LSTM baselines?

2. **RQ2:** After controlling for recurrent structure, hidden dimension, classical projections, total parameter count, data pipeline, optimization protocol, and random seed, does replacing the VQC by an exact classical surrogate materially change forecasting performance?
3. **RQ3:** Are the empirical effects of the VQC consistent across GRU and LSTM recurrent dynamics?
4. **RQ4:** How do the quantum and matched classical models differ in seed robustness, convergence behavior, and practical computational cost?

The principal contributions of this work are summarized as follows:

- We develop an architecture-controlled benchmarking framework that distinguishes the performance gains of the complete hybrid recurrent design from those attributable specifically to the VQC, using exact classical gate-core replacement controls.
- We conduct a paired ten-seed evaluation of QGRU and QLSTM using RMSE, MAE, R^2, seed-wise win counts, paired t-tests, and Wilcoxon signed-rank tests to assess both effect magnitude and consistency across initializations.
- We analyse 1,949 epoch-level records across six model configurations to characterize training and validation convergence, checkpoint quality, convergence efficiency, early stopping behaviour, generalization gaps, and simulator runtime.
- We show that QGRU achieves a 5.47% reduction in test RMSE and a lower mean best validation MSE than its exact classical surrogate, whereas QLSTM and its surrogate remain in the same predictive regime, with the surrogate exhibiting greater validation stability.
- We perform a reproducibility audit demonstrating that 656 duplicated QGRU/-GRU epoch records agree with the corresponding primary logs to within 5×10^{-7} absolute loss units.

2 Related Work

2.1 Classical recurrent forecasting

GRU and LSTM networks model temporal dependencies through gated recurrent state updates. The GRU employs update and reset gates to regulate information retention and state modification, whereas the LSTM introduces input, forget, and output gates together with a separate memory cell. Both architectures are widely used as nonlinear baselines for one-step and multi-step regression forecasting because they can represent long- and short-range temporal structure while mitigating vanishing-gradient effects. Accordingly, a rigorous evaluation of quantum recurrent models should consider not only compact recurrent networks with approximately matched parameter counts, but also stronger classical comparators that are matched in hidden-state width and tuned under the same validation protocol. Verified references on meteorological GRU/LSTM forecasting should be inserted here [1–4].

2.2 Variational quantum circuits and hybrid learning

Variational quantum circuits extend classical learning pipelines by combining data-dependent quantum state preparation, trainable parametrized gates, entangling operations, and quantum measurements. In hybrid architectures, the resulting expectation values are returned to a classical computation graph, transformed by classical readout layers, and optimized with respect to a classical objective. Prior studies have investigated VQCs in classification, regression, sequence modelling, and recurrent learning [6–8, 12–18]. These studies motivate the use of VQCs as compact nonlinear transformations, but performance improvements in hybrid models cannot be attributed to the quantum component unless the surrounding classical architecture is adequately controlled. The present work therefore does not presume either computational or predictive advantage; instead, it evaluates whether forecasting performance changes when the VQC is replaced by a parameter-matched classical nonlinear module.

2.3 Controlled quantum–classical benchmarking

A central methodological challenge in quantum machine learning is the construction of classical controls that are matched not only in total parameter count, but also in data access, representation width, optimization budget, and surrounding classical architecture. When these factors differ, an observed performance gain may arise from hidden-state capacity, projection layers, training conditions, or other architectural components rather than from the quantum circuit itself. Controlled benchmarking must therefore distinguish the contribution of the complete hybrid model from the contribution of the VQC embedded within it. The methodology adopted in this paper follows this stricter attribution principle [19–21].

3 Problem Formulation

Let

$$\mathbf{x}_t \in \mathbb{R}^d$$

denote the meteorological feature vector observed at time index t, where $d = 17$, and let

$$y_t \in \mathbb{R}$$

denote the target variable `2t`. Because the historical value of the target is included among the input variables, the forecasting task is formulated as an autoregressive problem with exogenous meteorological covariates.

For a sequence length of $L = 24$, each model receives the input window

$$\mathbf{X}_t = [\mathbf{x}_{t-L+1}, \mathbf{x}_{t-L+2}, \ldots, \mathbf{x}_t] \in \mathbb{R}^{L \times d}$$

and predicts the target value at the subsequent time step:

$$\widehat{y}_{t+1} = f_\Theta\left(\mathbf{X}_t\right),$$

where f_Θ denotes a QGRU, QLSTM, compact classical recurrent baseline, or an exact classical VQC-replacement control, and Θ represents the corresponding trainable model parameters.

The model parameters are learned by minimizing the mean squared error over the training set:

$$\mathcal{L}_{\mathrm{MSE}}(\Theta) = \frac{1}{N_{\mathrm{tr}}} \sum_{i=1}^{N_{\mathrm{tr}}} (y_i - \widehat{y}_i)^2 . \tag{1}$$

The central inferential objective is not limited to determining whether

$$\mathcal{E}(\text{quantum recurrent}) < \mathcal{E}(\text{compact classical recurrent}),$$

because this comparison does not isolate the source of any observed performance difference. Instead, the study evaluates whether

$$\mathcal{E}(\text{VQC gate core}) < \mathcal{E}(\text{exact classical gate core})$$

after the surrounding recurrent architecture is held fixed. This controlled formulation directly assesses whether the VQC contributes predictive value beyond that provided by the shared recurrent structure, hidden-state dimension, gate-specific classical projections, and optimization protocol.

4 Forecasting Data and Preprocessing

4.1 Input variables

The forecasting pipeline uses `merged_time_sorted.csv` to predict the two-metre air-temperature variable `2t`. The dataset comprises 1,445 chronologically ordered observations recorded between 1 January 2021 and 31 December 2024 at a single location in Delhi, India (28.6139° N, 77.2090° E). The data correspond to a historical ERA5 point series with a nominal daily temporal resolution, and the target variable `2t` is expressed in kelvin[5].

All numerical variables other than the target are used as exogenous covariates. The historical value of `2t` is additionally included as an autoregressive input, while no further target-lag variables are introduced. The resulting input representation therefore contains 16 exogenous variables and one historical target variable, yielding 17 input features in total. Table 1 lists the feature headers used in the reported experiments together with their physical interpretation.

Several numerical columns represent fixed spatial or observational metadata rather than time-varying meteorological quantities. Specifically, `lat`, `lon`, the four pressure-level variables, and the three measurement-height variables remain constant throughout the observation period. These nine zero-variance variables are retained to reproduce the reported experiments and are mapped to zero after standardization. The remaining eight variables provide the time-dependent information used by the forecasting models.

The original dataset schema contains two naming inconsistencies. The header `plev(UGRD-500mb))` includes an additional closing parenthesis, while the two

meridional-wind pressure-level headers contain trailing whitespace. These original column names were retained in the reported experiments to ensure consistency with the implemented pipeline. A future public release should provide a standardized schema together with an explicit mapping to the original headers to preserve reproducibility.

Table 1 Input variables used by the forecasting pipeline. The original column names are retained to reproduce the reported experimental configuration.

No.	Raw feature header	Role
1	`lat`	Latitude (28.6139° N); constant spatial metadata
2	`lon`	Longitude (77.2090° E); constant spatial metadata
3	`prmsl`	Mean sea-level pressure (Pa)
4	`2r`	Two-metre relative humidity (%)
5	`plev(UGRD-500mb))`	Constant pressure coordinate for 500-hPa zonal wind; the original header contains an additional parenthesis
6	`plev(UGRD-850mb)`	Constant pressure coordinate for 850-hPa zonal wind
7	`plev(VGRD-500mb)`	Constant pressure coordinate for 500-hPa meridional wind; the original header contains trailing whitespace
8	`plev(VGRD-850mb)`	Constant pressure coordinate for 850-hPa meridional wind; the original header contains trailing whitespace
9	`v(VGRD-500mb)`	Meridional wind component at 500 hPa ($\mathrm{m\,s^{-1}}$)
10	`v(VGRD-850mb)`	Meridional wind component at 850 hPa ($\mathrm{m\,s^{-1}}$)
11	`u(UGRD-10m)`	Zonal wind component at 10 m ($\mathrm{m\,s^{-1}}$)
12	`u(UGRD-500mb)`	Zonal wind component at 500 hPa ($\mathrm{m\,s^{-1}}$)
13	`u(UGRD-850mb)`	Zonal wind component at 850 hPa ($\mathrm{m\,s^{-1}}$)
14	`height(TMP-2m)`	Constant two-metre temperature-height metadata (m)
15	`height(UGRD-10m)`	Constant ten-metre wind-height metadata (m)
16	`height(RH-2m)`	Constant two-metre humidity-height metadata (m)
17	`2t`	Historical two-metre temperature (K) included as an autoregressive input

The dataset contains no missing numerical values, invalid timestamps, or duplicate timestamps. Accordingly, no observations are removed during required-variable filtering, and no imputation is performed. The temporal sequence is nevertheless not fully regular. Of the successive timestamp pairs, 1,438 are separated by one day, while six temporal discontinuities correspond to 16 absent calendar dates: one 11-day interval, one 3-day interval, and four 2-day intervals. The implications of these discontinuities for sequence construction are discussed below.

4.2 Chronological partitioning

The dataset is divided chronologically:

$$N_{\text{train}} = \lfloor 0.8N \rfloor, \qquad N_{\text{val}} = \lfloor 0.1N \rfloor, \qquad N_{\text{test}} = N - N_{\text{train}} - N_{\text{val}}. \tag{2}$$

For $N = 1{,}445$, this procedure assigns 1,156 observations to the training partition, 144 observations to the validation partition, and 145 observations to the test partition. The training interval extends from 1 January 2021 to 17 March 2024, the validation interval from 18 March 2024 to 8 August 2024, and the test interval from 9 August 2024 to 31 December 2024. This chronological ordering prevents future observations from contributing to model fitting for earlier periods.

Sequence construction is performed independently within each partition after the chronological split. With a sequence length of 24 observations, the resulting datasets contain 1,132 training windows, 120 validation windows, and 121 test windows. Each input window comprises observations $i, \ldots, i + 23$, and the corresponding label is the target at observation $i + 24$. Consequently, the first 24 observations of each partition are used only as historical context, and no sequence crosses a train–validation or validation–test boundary.

4.3 Scaling and leakage control

Separate feature and target scalers are fitted exclusively on the 1,156 observations in the training partition. The reported experiments use standard-score normalization. For a scalar training feature x,

$$x^{\text{scaled}} = \frac{x - \mu_{\text{train}}}{\sigma_{\text{train}}}. \tag{3}$$

The resulting training-derived means and standard deviations are then applied unchanged to the validation and test partitions. The target variable is standardized independently using statistics computed only from the training target. Before evaluation, both predictions and reference targets are transformed back to the original target scale; therefore, RMSE and MAE are reported in kelvin.

The preprocessing procedure incorporates two safeguards against information leakage. First, all normalization parameters are estimated solely from the training partition. Second, sequence generation is performed independently within each chronological partition, preventing validation or test observations from entering training windows and ensuring that no window crosses a partition boundary. Because the historical target is included as an input feature, the forecasting protocol assumes that the most recently observed temperature is available when each one-step-ahead prediction is generated.

4.4 Temporal continuity

The preprocessing framework supports segment-aware sequence construction in which timestamp differences are evaluated against an expected sampling interval and windows are prevented from crossing detected temporal discontinuities. This functionality was not activated in the reported experiments. The timestamp variable was therefore

excluded from the model inputs, and the complete dataset was processed as a single ordered sequence.

Consequently, the sequence length of 24 refers to 24 successive recorded observations rather than necessarily 24 consecutive calendar days. Because the dataset contains six identified temporal discontinuities, some input windows may span intervals containing unobserved calendar dates. This treatment reflects the experimental protocol used to obtain the reported results and does not introduce information from future observations; however, it may alter the effective temporal duration represented by individual windows.

Future evaluations should incorporate explicit timestamp-based continuity screening or an equivalent documented gap-handling procedure. Such an extension would ensure that every input window represents a consistent temporal duration and would allow the sensitivity of the reported findings to irregular sampling to be evaluated directly.

Table 2 Forecasting dataset and preprocessing protocol.

Item	**Configuration**
Data file	`merged_time_sorted.csv`
Dataset provenance	Historical ERA5 point series for Delhi
Spatial location	28.6139° N, 77.2090° E
Temporal coverage	1 January 2021–31 December 2024
Number of observations	1,445
Nominal sampling interval	Daily, with six temporal discontinuities corresponding to 16 absent dates
Prediction target	`2t`, two-metre temperature (K)
Number of input features	17
Time-varying input features	8
Constant metadata features	9
Historical target included	Yes
Additional target lags	None
Sequence length	24 successive recorded observations
Forecast horizon	One step ahead
Training partition	1,156 observations and 1,132 sequence windows
Validation partition	144 observations and 120 sequence windows
Test partition	145 observations and 121 sequence windows
Feature scaling	Standard-score normalization fitted exclusively on training observations
Target scaling	Standard-score normalization fitted exclusively on the training target
Missing numerical values	None; no imputation performed
Window construction	Performed independently within each chronological partition
Temporal-gap screening	Available in the preprocessing framework but not activated in the reported experiments

5 Hybrid Quantum Recurrent Models

5.1 Shared gate transformation

For each recurrent gate g, let

$$\mathbf{s}_{t,g} \in \mathbb{R}^{d+h}$$

denote the gate-specific state vector formed from the current input features and the relevant recurrent state. A learnable classical pre-projection compresses this vector to the four-dimensional quantum interface:

$$\mathbf{a}_{t,g} = \pi \tanh\left(W_g^{\text{pre}} \mathbf{s}_{t,g} + \mathbf{b}_g^{\text{pre}}\right) \in [-\pi, \pi]^4. \tag{4}$$

The hyperbolic tangent bounds the projected values before their use as quantum rotation angles, while the factor π maps them to the interval $[-\pi, \pi]$.

The encoded vector is then processed by an independent four-qubit VQC:

$$\mathbf{q}_{t,g} = \mathcal{Q}_g\left(\mathbf{a}_{t,g}; \Theta_g\right) = \Big[\langle Z_1\rangle \ \langle Z_2\rangle \ \langle Z_3\rangle \ \langle Z_4\rangle\Big]^\top. \tag{5}$$

The implemented circuit applies R_Y angle encoding to the four projected features and uses three variational layers. Each layer contains an independent three-parameter single-qubit rotation on every qubit followed by a ring of controlled-NOT entangling gates. The input angles are re-uploaded before the second and third variational layers. Measurement of the Pauli-Z expectation value on each qubit produces the four-dimensional quantum feature vector $\mathbf{q}_{t,g}$. Consequently, each gate-specific VQC contains $3 \times 4 \times 3 = 36$ trainable circuit parameters[6, 8].

A learnable post-projection expands the measured quantum features to the recurrent hidden dimension $h = 16$:

$$\mathcal{G}_g(\mathbf{s}_{t,g}) = W_g^{\text{post}} \mathbf{q}_{t,g} + \mathbf{b}_g^{\text{post}}. \tag{6}$$

Each recurrent gate contains its own pre-projection, variational parameters, and post-projection. Thus, gates share the same circuit topology but do not share trainable parameters. The circuits are evaluated using noiseless statevector simulation with analytic Pauli-Z expectation values and backpropagation-based differentiation.

5.2 Quantum–classical gate interface

Let $\mathbf{s}_{t,g} \in \mathbb{R}^{m_g}$ denote the classical state supplied to gate g. For the update, reset, forget, input, output, and LSTM candidate gates, $m_g = d+h = 33$ under the 17-feature, 16-hidden-unit configuration. The QGRU candidate gate uses

$$\mathbf{s}_{t,h} = \mathbf{x}_t \parallel (\mathbf{r}_t \odot \mathbf{h}_{t-1}),$$

which has the same dimensionality but incorporates the reset-modulated hidden state and therefore represents a different state-dependent input geometry.

The pre-projection

$$\mathbf{u}_{t,g} = W_g^{\text{pre}} \mathbf{s}_{t,g} + \mathbf{b}_g^{\text{pre}} \in \mathbb{R}^4$$

serves as more than a dimensionality-matching operation. It learns a gate-specific four-dimensional representation of the joint input–memory state before quantum encoding. The bounded transformation

$$\mathbf{a}_{t,g} = \pi \tanh(\mathbf{u}_{t,g})$$

restricts the rotation angles to a compact domain, limits periodic aliasing caused by unbounded classical activations, and preserves smooth gradients around the origin.

For a gate-specific VQC unitary $U_g(\mathbf{a}_{t,g}, \Theta_g)$, the quantum feature vector is

$$q_{t,g,j} = \langle 0|^{\otimes 4} U_g^\dagger(\mathbf{a}_{t,g}, \Theta_g) Z_j U_g(\mathbf{a}_{t,g}, \Theta_g) |0\rangle^{\otimes 4}, \qquad j = 1, \ldots, 4. \tag{7}$$

Hence $\mathbf{q}_{t,g} \in [-1, 1]^4$. The post-projection

$$\mathbf{v}_{t,g} = W_g^{\text{post}} \mathbf{q}_{t,g} + \mathbf{b}_g^{\text{post}} \in \mathbb{R}^{16}$$

maps the four measured observables back to the hidden-state coordinates required by the recurrent update. The resulting quantum–classical gate transformation is

$$\mathcal{G}_g(\mathbf{s}) = W_g^{\text{post}} \mathcal{Q}_g\big(\pi \tanh(W_g^{\text{pre}} \mathbf{s} + \mathbf{b}_g^{\text{pre}}); \Theta_g\big) + \mathbf{b}_g^{\text{post}}. \tag{8}$$

Equation (8) also establishes the basis of the controlled attribution analysis. A compact vanilla recurrent model does not isolate the contribution of the VQC because it omits the gate-specific compression to four bounded latent features and their subsequent expansion to the hidden-state space. The exact classical controls preserve the recurrent equations, hidden dimension, gate-specific pre-projections, post-projections, and overall parameter count, while replacing only $\mathcal{Q}_g$ with an equally parameterized classical nonlinear core. The central gate core is therefore the sole intended intervention, enabling any observed performance difference to be attributed specifically to the VQC rather than to differences in recurrent width or surrounding classical architecture.

5.3 VQC design

The VQC comprises four qubits and three variational layers. The bounded feature vector from Eq. (4) is first encoded through qubit-wise R_Y rotations. Each variational layer then applies a trainable `Rot` gate to every qubit, followed by a ring of CNOT entanglers. For qubit j in variational layer ℓ,

$$\text{Rot}\,(\theta_{\ell j1}, \theta_{\ell j2}, \theta_{\ell j3}) = R_Z(\theta_{\ell j1}) R_Y(\theta_{\ell j2}) R_Z(\theta_{\ell j3}). \tag{9}$$

The entangling operation follows the ring topology

$$\text{CNOT}(1,2),\ \text{CNOT}(2,3),\ \text{CNOT}(3,4),\ \text{CNOT}(4,1).$$

Consistent with the implementation, the bounded angle vector is initially encoded before the first variational layer and re-uploaded before the second and third layers. This repeated encoding allows the same four-dimensional gate-specific input to interact with the trainable circuit parameters at multiple circuit depths. The circuit output is formed by measuring the Pauli-Z expectation value of each qubit.

Figure 1 illustrates the repeated four-qubit circuit template used by the recurrent gates. The reported implementation contains three variational layers, each comprising data-dependent R_Y encoding where applicable, trainable Rot operations, ring-CNOT entanglement, and final Pauli-Z measurement. Although the recurrent gates share this circuit topology, each gate is instantiated with an independent trainable parameter tensor[7].

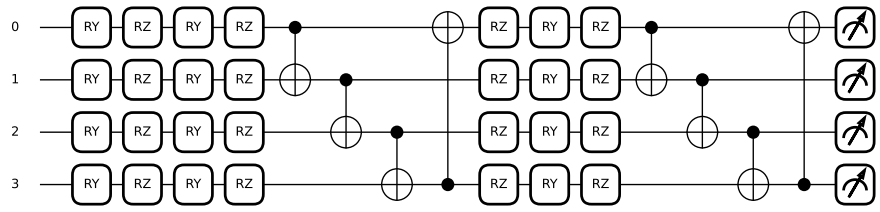


Fig. 1 Four-qubit variational quantum circuit used in each QGRU gate. The gate-specific 4-dimensional latent vector is encoded through bounded rotation angles, processed by three trainable variational layers with ring-CNOT entanglement, and measured through four Pauli-Z expectation values. The three QGRU gates use the same topology but independent circuit parameters.

Because each rotation gate contains three trainable angles, the number of trainable VQC parameters per recurrent gate is

$$P_{\mathrm{VQC,gate}} = L_q n_q P_{\mathrm{Rot}} = 3 \times 4 \times 3 = 36. \tag{10}$$

The experiments use PennyLane's noiseless statevector simulator with backpropagation [9]. The circuit returns analytic expectation values; no finite-shot sampling, device-noise model, or physical QPU is used.

5.4 Classical recurrent reference architectures

For architectural context, Figs. 2 and 3 illustrate the conventional LSTM and GRU cells from which the corresponding hybrid quantum recurrent models are derived. These diagrams are included solely to clarify the recurrent information flow and the roles of the gating mechanisms. They do not represent the classical models used as width-matched controls in the experiments. Instead, the compact GRU and LSTM baselines are selected under the approximately equal total-parameter regime summarized in Table 3.

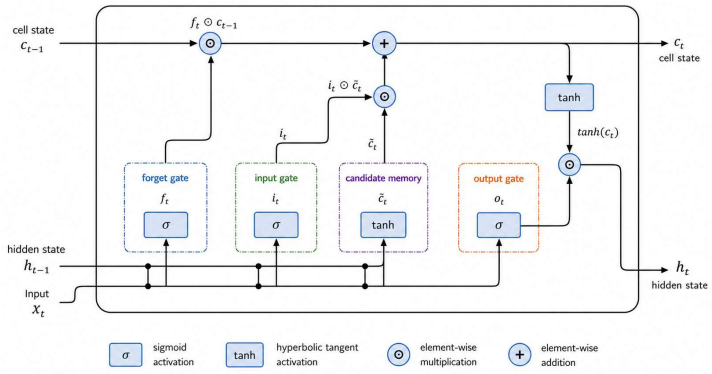


Fig. 2 Classical LSTM reference architecture. The forget, input, candidate-memory, and output transformations regulate the persistent cell state $\mathbf{c}_t$ and the exposed hidden state $\mathbf{h}_t$.

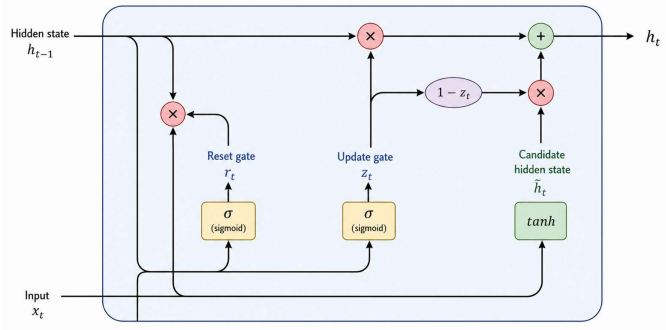


Fig. 3 Classical GRU reference architecture. The update and reset gates regulate the combination of the previous hidden state and the candidate state without introducing a separate memory cell.

5.5 QGRU formulation

The QGRU comprises a single recurrent layer with hidden state $\mathbf{h}_t \in \mathbb{R}^{16}$ and three independent quantum–classical gate modules. At each time step, the update and reset gates receive the concatenation of the current input $\mathbf{x}_t$ and the previous hidden state $\mathbf{h}_{t-1}$. The candidate-state gate instead receives the current input together with the reset-modulated hidden state. Using the gate transformation $\mathcal{G}_g$ defined above, the recurrent update is

$$\mathbf{z}_t = \sigma\left(\mathcal{G}_z\left(\mathbf{x}_t \parallel \mathbf{h}_{t-1}\right)\right), \tag{11}$$

$$\mathbf{r}_t = \sigma\left(\mathcal{G}_r\left(\mathbf{x}_t \parallel \mathbf{h}_{t-1}\right)\right), \tag{12}$$

$$\widetilde{\mathbf{h}}_t = \tanh\left(\mathcal{G}_h\left(\mathbf{x}_t \parallel \left(\mathbf{r}_t \odot \mathbf{h}_{t-1}\right)\right)\right), \tag{13}$$

$$\mathbf{h}_t = \left(1 - \mathbf{z}_t\right) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \widetilde{\mathbf{h}}_t. \tag{14}$$

Here, $\mathbf{z}_t$ controls the interpolation between the previous hidden state and the candidate state, whereas $\mathbf{r}_t$ regulates the contribution of the previous hidden state to candidate-state construction. Each of the update, reset, and candidate transformations uses an independent pre-projection, four-qubit VQC, and post-projection; no circuit parameters are shared across the three [2, 15, 16].

After the complete input sequence has been processed, the final hidden state is mapped to the one-step-ahead scalar prediction through a linear output head:

$$\widehat{y}_{t+1} = \mathbf{w}_y^{\top} \mathbf{h}_t + b_y. \tag{15}$$

The complete QGRU contains 773 trainable parameters. Figure 4 summarizes the corresponding recurrent information flow.

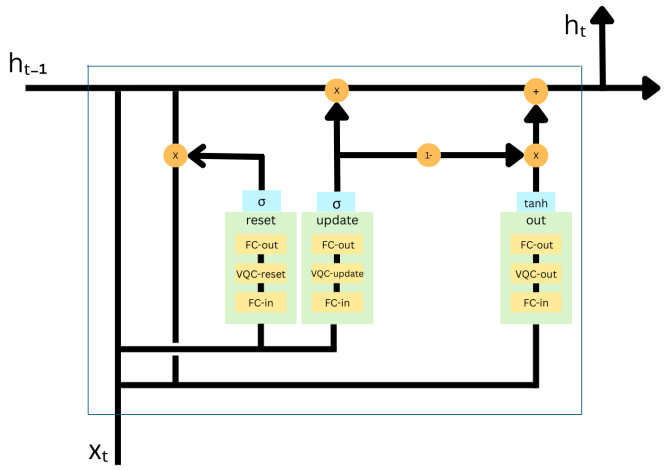


Fig. 4 QGRU architecture used in the controlled benchmark. The current input and previous hidden state are combined separately for the update, reset, and candidate branches. Each branch applies a gate-specific classical compression, four-qubit VQC, and classical expansion before the GRU state update. The exact control preserves this entire computational graph and replaces only the VQC core.

5.6 QLSTM formulation

The QLSTM comprises a single recurrent layer with hidden state $\mathbf{h}_t \in \mathbb{R}^{16}$, cell state $\mathbf{c}_t \in \mathbb{R}^{16}$, and four independent quantum–classical gate modules. At each time step, the current input and previous hidden state are concatenated to form the common gate input

$$\mathbf{s}_t = \mathbf{x}_t \parallel \mathbf{h}_{t-1}.$$

The forget, input, and output gates apply sigmoid activations to their respective gate transformations, while the candidate-memory branch uses a hyperbolic tangent activation. The resulting QLSTM recurrence is

$$\mathbf{f}_t = \sigma\left(\mathcal{G}_f(\mathbf{s}_t)\right), \tag{16}$$
$$\mathbf{i}_t = \sigma\left(\mathcal{G}_i(\mathbf{s}_t)\right), \tag{17}$$
$$\mathbf{o}_t = \sigma\left(\mathcal{G}_o(\mathbf{s}_t)\right), \tag{18}$$
$$\widetilde{\mathbf{c}}_t = \tanh\left(\mathcal{G}_c(\mathbf{s}_t)\right), \tag{19}$$
$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \widetilde{\mathbf{c}}_t, \tag{20}$$
$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t), \tag{21}$$
$$\widehat{y}_{t+1} = \mathbf{w}_y^\top \mathbf{h}_t + b_y. \tag{22}$$

The forget gate $\mathbf{f}_t$ determines how much of the previous cell state is retained, the input gate $\mathbf{i}_t$ regulates the incorporation of the candidate memory $\widetilde{\mathbf{c}}_t$, and the output gate $\mathbf{o}_t$ controls the portion of the updated cell state exposed through $\mathbf{h}_t$. The additive cell-state pathway remains identical to the conventional LSTM recurrence; only the nonlinear transformations used to construct the four gate vectors are implemented

through independent quantum–classical modules. Figure 5 summarizes this recurrent structure[1, 13].

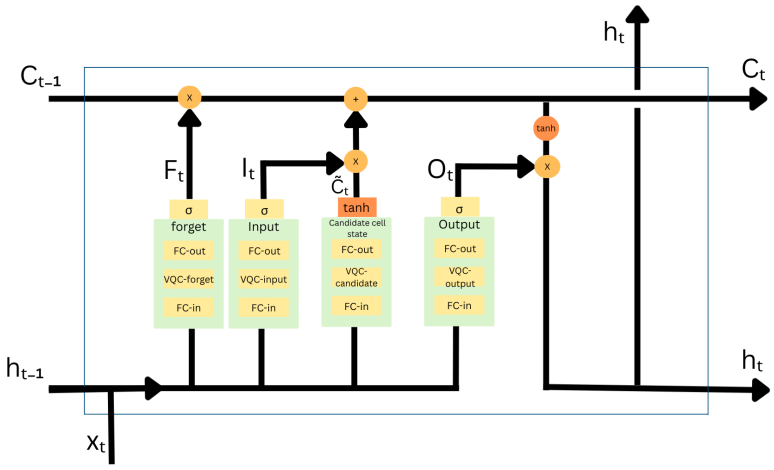


Fig. 5 QLSTM architecture used in the controlled benchmark. Independent quantum–classical modules compute the forget, input, candidate-memory, and output transformations, while the additive cell-state pathway retains the standard LSTM semantics. The exact classical control changes only the four VQC cores and preserves every surrounding projection and recurrent operation.

Each QLSTM gate uses an independent pre-projection, VQC parameter tensor, and post-projection. Consequently, the four gates share only the circuit topology and not their trainable parameters. In the exact classical control, the recurrent equations, hidden and cell dimensions, gate-specific projections, and scalar output head are preserved; only the four VQC cores are replaced by parameter-matched classical nonlinear transformations. The complete QLSTM contains 1,025 trainable parameters. Figure 6 shows the common four-qubit circuit topology used independently by the four gates.

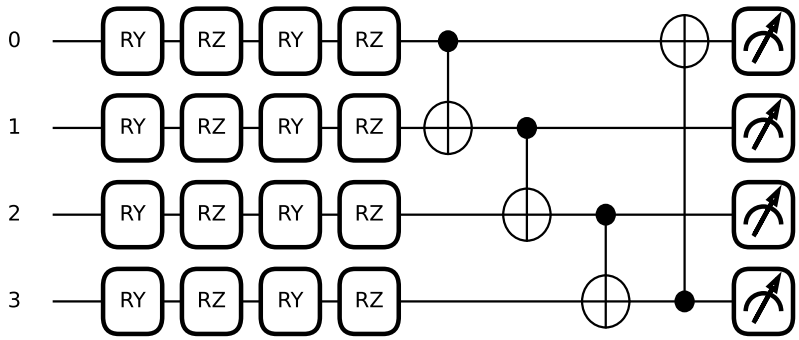


Fig. 6 Four-qubit variational quantum circuit used in each QLSTM gate. The forget, input, candidate-memory, and output gates share the circuit topology but use independent pre-projections, variational parameters, and post-projections. This separation permits each gate to learn a distinct nonlinear transformation while preserving an exactly controlled 36-parameter quantum core per gate.

6 Classical Baselines and Exact Controls

6.1 Compact parameter-matched GRU and LSTM

Two compact classical recurrent models are included to determine whether the complete hybrid architectures use a comparable trainable-parameter budget more effectively than conventional recurrent networks. Both baselines consist of a standard single-layer PyTorch recurrent module followed by a scalar linear output [24]. Their hidden dimensions are selected to approximate the total parameter counts of the corresponding quantum models:

- compact classical GRU: hidden size 9, 766 parameters;
- compact classical LSTM: hidden size 9, 1,018 parameters.

These comparisons address the architecture-level question:

Which architecture uses approximately the same total trainable-parameter budget most effectively?

The compact baselines provide an important measure of overall parameter efficiency, but they cannot identify the source of any observed performance difference. In particular, they differ from the quantum models in hidden-state width, internal gate parameterization, and the absence of gate-specific compression and expansion layers. Consequently, superiority over a compact GRU or LSTM may demonstrate the effectiveness of the complete hybrid architecture, but it cannot by itself be interpreted as evidence for a VQC-specific contribution.

6.2 Exact classical VQC-replacement QGRU control

A second and more stringent comparison is introduced to isolate the contribution of the VQC within the QGRU. The exact classical control preserves:

- the same three-gate GRU recurrence;
- hidden dimension $h = 16$;
- independent gate-specific pre- and post-projections;
- the same 24-step inputs, output head, optimizer, scheduler, early stopping, and seeds;
- the same total parameter count of 773.

The only architectural modification is the replacement of the VQC mapping in Eq. (5) by the classical residual gate core

$$\mathrm{Core}(\mathbf{u}) = \mathbf{u} + \boldsymbol{\gamma} \odot \tanh\left(\mathrm{LayerNorm}(W\mathbf{u} + \mathbf{b})\right) + \boldsymbol{\beta}, \tag{23}$$

where $\mathbf{u} \in \mathbb{R}^4$. This four-dimensional nonlinear transformation combines a learned linear map, layer normalization, a hyperbolic tangent activation, and trainable element-wise residual gain and bias terms[23]. Its trainable parameter count is

$$P_{\mathrm{Linear}(4,4)} = 4 \times 4 + 4 = 20, \tag{24}$$
$$P_{\mathrm{LayerNorm}(4)} = 4 + 4 = 8, \tag{25}$$
$$P_{\boldsymbol{\gamma}} = 4, \tag{26}$$
$$P_{\boldsymbol{\beta}} = 4, \tag{27}$$
$$P_{\mathrm{core}} = 20 + 8 + 4 + 4 = 36. \tag{28}$$

Thus,

$$P_{\mathrm{core}} = P_{\mathrm{VQC,gate}} = 36.$$

The replacement core therefore matches the 36 trainable parameters of each three-layer, four-qubit VQC exactly. Because the QGRU contains three independent gate modules, preserving the parameter count at the gate level also preserves the complete model size. The resulting classical control contains exactly 773 trainable parameters, equal to the QGRU.

This comparison is designed as a controlled gate-core intervention. The recurrent equations, hidden-state dimension, gate count, classical pre-projections, classical post-projections, output head, input sequences, and optimization protocol remain unchanged. Therefore, the central four-dimensional transformation is the only intended difference between the two models. A performance difference between the QGRU and this control can consequently be associated with replacing the classical gate core by the VQC, rather than with additional parameters, a wider recurrent state, or a different surrounding architecture.

6.3 Exact classical VQC-replacement QLSTM control

The exact QLSTM control applies the same attribution principle to the four-gate LSTM architecture. It preserves the 16-dimensional hidden state, 16-dimensional cell state, forget, input, output, and candidate-memory gates, gate-specific pre- and post-projections, scalar output head, training protocol, and total model size. Only the VQC within each gate is replaced by the two-layer classical nonlinear transformation

$$\mathbf{u} \mapsto \tanh\left(W_2 \tanh(W_1\mathbf{u} + \mathbf{b}_1)\right), \tag{29}$$

with

$$W_1 \in \mathbb{R}^{4\times 4}, \quad \mathbf{b}_1 \in \mathbb{R}^4, \quad W_2 \in \mathbb{R}^{4\times 4},$$

where the second layer has no bias. Its parameter count is

$$(4 \times 4 + 4) + (4 \times 4) = 20 + 16 = 36$$

per gate, exactly matching the three-layer four-qubit VQC. The complete QLSTM surrogate therefore contains exactly 1,025 trainable parameters.

As in the QGRU control, the four-dimensional gate core is the sole intended intervention. The standard LSTM state equations, additive cell-state pathway, hidden and cell dimensions, and all surrounding classical transformations are retained. The resulting comparison therefore tests whether the VQC provides predictive value beyond that of an equally parameterized classical nonlinear transformation embedded within the same QLSTM architecture.

Table 3 Role of the two classical comparison regimes.

Comparison	Controlled quantity	Question answered
Quantum model vs. compact vanilla recurrent model	Approximately equal total trainable parameters	Is the complete hybrid architecture more parameter-efficient than a small conventional recurrent baseline?
Quantum model vs. exact classical gate-core surrogate	Same recurrence, width, projections, gate count, training protocol, and exact total parameters	Does the VQC itself produce a measurable predictive contribution?

7 Experimental Protocol

7.1 Training configuration

All quantum and classical models are trained under an identical optimization protocol to ensure that performance differences are not attributable to unequal training conditions. Model parameters are optimized using Adam with mean squared error (MSE) as the training objective. The initial learning rate is 10^{-3}, the weight-decay coefficient is 10^{-4},

and the global gradient norm is clipped at 1.0 to limit excessively large parameter updates[22].

The learning rate is reduced by a factor of 0.5 when the validation loss reaches a plateau. Model selection and early stopping are based exclusively on validation MSE. Training is terminated after eight consecutive epochs without validation improvement or after a maximum of 40 epochs. The parameter checkpoint attaining the lowest validation MSE is restored before evaluation on the held-out test partition. All models use the same batch size, random seeds, optimization settings, stopping criterion, and checkpoint-selection procedure.

Table 4 Shared training configuration used for all quantum and classical models.

Hyperparameter	**Value**
Random seeds	0–9
Batch size	32
Maximum epochs	40
Optimizer	Adam
Initial learning rate	10^{-3}
Weight decay	10^{-4}
Training objective	Mean squared error
Gradient clipping	Maximum norm 1.0
Learning-rate adaptation	Reduction on validation-loss plateau
Learning-rate reduction factor	0.5
Early-stopping criterion	Validation MSE
Early-stopping patience	8 epochs
Checkpoint selection	Minimum validation-MSE checkpoint
Quantum simulation	Noiseless statevector simulation
Differentiation	Backpropagation
Quantum measurements	Analytic expectation values

The variational circuits are evaluated using PennyLane's noiseless statevector simulator. Circuit outputs are computed as analytic expectation values and differentiated through the hybrid computational graph using backpropagation. No finite-shot sampling, device-noise model, or physical quantum processor is employed. The resulting runtime measurements therefore characterize the simulator-based implementations rather than quantum-hardware [9].

Table 5 summarizes the recurrent widths, quantum-circuit configurations, parameter counts, and gate-core constructions used in the controlled benchmark.

Table 5 Model architectures and gate-core configurations.

Model	Hidden	Qubits	VQC layers	Parameters	Gate core
QGRU	16	4	3	773	Independent VQC for each gate
Exact QGRU classical control	16	–	–	773	36-parameter residual classical core
Compact GRU	9	–	–	766	Standard GRU transformations
QLSTM	16	4	3	1,025	Independent VQC for each gate
Exact QLSTM classical control	16	–	–	1,025	36-parameter two-layer classical core
Compact LSTM	9	–	–	1,018	Standard LSTM transformations

7.2 Evaluation metrics

Predictive performance is evaluated after transforming the model outputs and reference targets back to the original target scale. Given N test targets y_i and corresponding predictions $\widehat{y}_i$, the reported metrics are

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \widehat{y}_i)^2}, \tag{30}$$

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}\left|y_i - \widehat{y}_i\right|, \tag{31}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \widehat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}. \tag{32}$$

RMSE is treated as the primary predictive metric because it quantifies error in the original target units and places greater emphasis on larger deviations. MAE provides a complementary measure of the typical absolute prediction error, whereas R^2 measures the proportion of test-set variance explained by the model. Because these metrics are calculated from the same predictions, they are interpreted as complementary summaries rather than independent statistical outcomes.

For each model and metric, results are summarized over $S = 10$ random seeds using the arithmetic mean and sample standard deviation:

$$\bar{m} = \frac{1}{S}\sum_{s=1}^{S} m_s, \tag{33}$$

$$s_m = \sqrt{\frac{1}{S-1}\sum_{s=1}^{S}(m_s - \bar{m})^2}. \tag{34}$$

The mean characterizes average predictive performance, while the sample standard deviation quantifies sensitivity to parameter initialization and stochastic optimization.

7.3 Paired statistical analysis

All model pairs are evaluated using the same seed labels. Statistical comparisons are therefore performed on paired seed-wise outcomes rather than independent samples. This design reduces the influence of between-seed variability and directly quantifies the performance change observed under matched initialization labels.

For an error metric such as RMSE, the paired difference is defined as

$$d_s = m_{s,\text{control}} - m_{s,\text{quantum}},$$

so $d_s > 0$ favors the quantum model. For R^2, the direction is defined as

$$d_s = R^2_{s,\text{quantum}} - R^2_{s,\text{control}}.$$

Thus, positive differences consistently indicate superior performance by the quantum model across all reported metrics.

Two-sided paired t-tests assess whether the mean paired difference departs from zero without specifying a directional alternative. One-sided Wilcoxon signed-rank tests provide a complementary non-parametric assessment of whether the paired differences systematically favor the quantum model. The Wilcoxon test does not require normally distributed paired differences, although it assumes that the nonzero differences arise from a symmetric distribution.

Because each comparison contains only ten paired observations, the tests have limited statistical power. The results are therefore interpreted using the combined evidence from effect magnitude, paired-test outcomes, seed-wise win counts, variability across seeds, and consistency across RMSE, MAE, and R^2. Statistical conclusions are not based solely on whether an individual p-value crosses a fixed threshold.

7.4 Convergence and optimization diagnostics

Epoch-level training records are analysed to characterize convergence, checkpoint quality, optimization efficiency, train–validation separation, and simulator runtime. For every model and seed, the recorded quantities include training loss, validation loss, elapsed epoch time, learning rate, and checkpoint status.

Let $\mathcal{L}^{\text{tr}}_{m,s,e}$ and $\mathcal{L}^{\text{val}}_{m,s,e}$ denote the scaled MSE for model m, seed s, and epoch e. The mean validation trajectory is computed as

$$\overline{\mathcal{L}}^{\text{val}}_{m,e} = \frac{1}{|\mathcal{S}_{m,e}|}\sum_{s\in\mathcal{S}_{m,e}} \mathcal{L}^{\text{val}}_{m,s,e}, \tag{35}$$

where $\mathcal{S}_{m,e}$ contains the seeds that remain active at epoch e. Because early stopping produces trajectories of unequal length, late-epoch averages may otherwise be determined by only a small number of runs. An epoch is therefore displayed only when at least 50% of the ten seeds contribute to the corresponding average.

Since the initial validation losses differ across model architectures, normalized validation convergence is defined as

$$\widetilde{\mathcal{L}}_{m,e} = \frac{\overline{\mathcal{L}}^{\mathrm{val}}_{m,e} - \min_j \overline{\mathcal{L}}^{\mathrm{val}}_{m,j}}{\overline{\mathcal{L}}^{\mathrm{val}}_{m,1} - \min_j \overline{\mathcal{L}}^{\mathrm{val}}_{m,j}}. \tag{36}$$

Thus, $\widetilde{\mathcal{L}}_{m,1} = 1$, while the best mean validation value is mapped to zero. This normalization describes the fraction of the attainable validation improvement remaining at each epoch and facilitates comparison of convergence profiles across architectures with different absolute loss levels.

For each seed, the convergence-efficiency epoch e_{95} is defined as the first epoch satisfying

$$\mathcal{L}^{\mathrm{val}}_{m,s,e} \le \mathcal{L}^{\mathrm{val},\star}_{m,s} + 0.05\left(\mathcal{L}^{\mathrm{val}}_{m,s,1} - \mathcal{L}^{\mathrm{val},\star}_{m,s}\right), \tag{37}$$

where $\mathcal{L}^{\mathrm{val},\star}_{m,s}$ is the minimum validation loss attained by model m under seed s. Accordingly, e_{95} identifies the earliest epoch at which 95% of the total observed validation improvement has been achieved.

The separation between the mean validation and training trajectories is summarized by

$$G_{m,e} = \overline{\mathcal{L}}^{\mathrm{val}}_{m,e} - \overline{\mathcal{L}}^{\mathrm{tr}}_{m,e}. \tag{38}$$

A positive value indicates that validation loss exceeds training loss at epoch e. This quantity is used as a descriptive measure of train–validation separation and is not treated as a substitute for held-out test performance.

Runtime is summarized using the median epoch time across seeds. The median is used because the execution logs contain occasional interruptions and because the fastest classical model is affected by coarse timing resolution. These measurements describe the observed statevector-simulation environment and are not interpreted as evidence of quantum computational speedup or hardware advantage.

The convergence and optimization statistics are treated as exploratory secondary analyses. Their p-values are interpreted together with the associated effect magnitudes and observed trajectory patterns rather than as independent confirmatory tests.

7.5 Log-source integrity and duplicate handling

The primary convergence analysis uses one designated source log for each of the six evaluated models: QGRU, exact QGRU control, compact GRU, QLSTM, exact QLSTM control, and compact LSTM. An additional log reproduces the QGRU-versus-compact-GRU experiment. This repeated record is excluded from all aggregate statistics and statistical comparisons to prevent duplicate observations from being counted more than once.

The repeated log is retained exclusively as a numerical-integrity audit. Across 656 matched model–seed–epoch records, the maximum absolute discrepancy between the

primary and repeated logs is 5×10^{-7} for both training and validation loss. This discrepancy is consistent with decimal-formatting precision rather than a substantive difference between the recorded optimization trajectories. The audit therefore verifies numerical consistency while preserving the independence of the primary aggregate analysis.

8 Results and Core Empirical Analysis

This section presents the empirical evidence in the order required to address the central research questions. We first examine held-out predictive performance under comparable parameter budgets, then use the exact gate-core controls to attribute any improvement specifically to the VQC. The subsequent analyses investigate whether the predictive ordering is supported by convergence behaviour, checkpoint quality, seed robustness, early-stopping dynamics, generalization, and computational cost. Finally, the source-log audit and complete seed-wise results establish the numerical integrity and reproducibility of the reported comparisons. The analysis uses one primary source log for each of the six models and contains 1,949 epoch-level records. The repeated QGRU/compact-GRU log is used exclusively for the numerical-integrity audit in Section 8.15 and is excluded from all model averages.

8.1 Consolidated held-out predictive performance

Table 6 provides the primary held-out comparison over ten random seeds. Under approximately equal total parameter budgets, both hybrid recurrent architectures outperform their compact vanilla counterparts across all three metrics. Within the GRU family, QGRU reduces mean RMSE from 1.3637 to 1.1045, corresponding to a 19.01% improvement, and reduces MAE by 20.76%. Its mean R^2 also increases from 0.94889 to 0.96821. Within the LSTM family, QLSTM reduces mean RMSE from 1.3931 to 1.0476, a 24.80% improvement, and reduces MAE by 24.29%, while increasing mean R^2 from 0.94944 to 0.97143.

Table 6 Held-out test performance over ten seeds. Values are reported as mean ± sample standard deviation. Lower RMSE and MAE indicate better performance, whereas higher R^2 indicates better predictive accuracy.

Family	Model	Parameters	RMSE	MAE	R^2
GRU	QGRU	773	1.1045 ± 0.0693	0.8354 ± 0.0540	0.96821 ± 0.00397
	Exact classical gate-core control	773	1.1684 ± 0.0863	0.8874 ± 0.0602	0.96437 ± 0.00528
	Compact classical GRU	766	1.3637 ± 0.3472	1.0543 ± 0.3259	0.94889 ± 0.02936
LSTM	QLSTM	1,025	1.0476 ± 0.0543	0.7906 ± 0.0503	0.97143 ± 0.00305
	Exact classical gate-core control	1,025	1.0603 ± 0.0327	0.7841 ± 0.0270	0.97078 ± 0.00180
	Compact classical LSTM	1,018	1.3931 ± 0.0814	1.0442 ± 0.0600	0.94944 ± 0.00596

These results establish that the complete hybrid architectures are more effective than the compact recurrent baselines under the tested parameter budgets. They do

not, however, isolate the source of the improvement because the compact models use a smaller hidden state and omit the gate-specific compression and expansion maps. The comparison therefore supports an architecture-level benefit, but not yet a VQC-specific conclusion.

The dispersion reported in Table 6 further distinguishes accuracy from robustness. The compact GRU exhibits substantially greater seed-to-seed variability than the remaining models, largely because of its poor runs at seeds 7 and 8. Its mean disadvantage therefore reflects both higher typical error and greater sensitivity to initialization. The compact LSTM is more stable than the compact GRU, but it remains inferior to both 16-state LSTM-family models for every evaluated seed and metric. By contrast, the quantum models and their exact controls occupy narrower and substantially stronger predictive regimes.

8.2 Paired attribution of the VQC gate core

The exact-control comparisons provide the decisive attribution test because they preserve the recurrent equations, hidden width, gate count, gate-specific classical projections, optimization protocol, and total parameter count. Consequently, the gate-core transformation is the sole intended intervention.

Within the GRU family, QGRU reduces mean RMSE by 0.06394 relative to its exact 773-parameter surrogate, corresponding to a 5.47% reduction. It also reduces MAE by 0.05195, or 5.85%, and increases R^2 by 0.00384. QGRU performs better in seven of the ten paired seeds for each metric. As reported in Table 7, the two-sided paired tests remain above 0.05, whereas the directional Wilcoxon test for MAE is below 0.05 and the corresponding RMSE and R^2 tests lie close to that threshold. The consistency of the effect direction across all three metrics, together with the seven-of-ten seed advantage, supports a coherent and practically meaningful QGRU VQC effect. Nevertheless, the small number of paired observations limits the strength of the statistical evidence, and the result is therefore interpreted as moderately supported rather than definitive.

Table 7 Paired analysis of the exact VQC-replacement controls. For RMSE and MAE, positive mean difference denotes lower error for the quantum model; for R^2, it denotes higher explained variance. p_t is the two-sided paired t-test and p_W is the directional Wilcoxon signed-rank test used in the original evaluation.

Pair	Metric	Mean difference	Quantum wins	p_t	p_W
QGRU/control	RMSE	0.06394	7/10	0.1004	0.0527
QGRU/control	MAE	0.05195	7/10	0.0991	0.0420
QGRU/control	R^2	0.00384	7/10	0.0988	0.0527
QLSTM/control	RMSE	0.01271	7/10	0.502	0.216
QLSTM/control	MAE	−0.00651	4/10	0.661	0.688
QLSTM/control	R^2	0.00065	7/10	0.535	0.216

The QLSTM comparison yields a different attribution pattern. QLSTM has a small mean advantage of 0.01271 in RMSE and 0.00065 in R^2, while the exact control

has a small MAE advantage of 0.00651. None of these paired differences is statistically persuasive. Moreover, the direction of the MAE result differs from those of RMSE and R^2. QLSTM and its exact surrogate therefore occupy the same practical predictive regime. The substantial QLSTM-versus-compact-LSTM improvement is consequently attributable primarily to the wider recurrent architecture and its gate-specific projections rather than to a separately identifiable VQC effect.

Taken together, the held-out and paired results establish two distinct findings. The complete hybrid architectures outperform compact parameter-matched recurrent models, but the VQC-specific contribution depends on the recurrent family. Evidence for a VQC contribution is directionally coherent in QGRU, whereas QLSTM remains practically indistinguishable from its exact classical gate-core surrogate.

8.3 Complete seed-wise test results

Tables 8 and 9 report every seed-level test result used in the aggregate statistics and paired comparisons. These complete outcomes make the model rankings auditable and reveal initialization sensitivity, occasional performance reversals, and unstable runs that are hidden by aggregate means.

Within the GRU family, QGRU achieves the best RMSE in seven of ten seeds, the exact classical control in two, and the compact GRU in one. The compact GRU is competitive at seed 0, but its pronounced failures at seeds 7 and 8 substantially increase both its mean error and variance. QGRU therefore provides a more robust result across initializations even though its advantage over the exact control is not universal.

Table 8 Complete seed-wise GRU-family test results. Lower RMSE and MAE are better; higher R^2 is better.

	QGRU			**Exact control**			**Compact GRU**		
Seed	**RMSE**	**MAE**	R^2	**RMSE**	**MAE**	R^2	**RMSE**	**MAE**	R^2
0	1.136868	0.874541	0.966434	1.156085	0.890822	0.965290	1.118728	0.844915	0.967497
1	1.098653	0.835046	0.968653	1.175138	0.903165	0.964137	1.296831	0.984180	0.956324
2	1.215341	0.926335	0.961641	1.086145	0.831082	0.969363	1.274766	0.940692	0.957798
3	0.988668	0.744730	0.974615	1.243013	0.980171	0.959874	1.140420	0.857821	0.966224
4	1.042465	0.784151	0.971777	1.028075	0.798647	0.972551	1.252916	0.950113	0.959232
5	1.152342	0.856841	0.965515	1.331927	0.974497	0.953928	1.166943	0.888941	0.964635
6	1.104262	0.822527	0.968332	1.095436	0.817630	0.968836	1.162881	0.859642	0.964881
7	1.035605	0.792614	0.972148	1.160474	0.890307	0.965026	2.164138	1.830463	0.878369
8	1.092562	0.826669	0.969000	1.213643	0.902532	0.961748	1.825963	1.456681	0.913412
9	1.178046	0.890949	0.963959	1.194249	0.885028	0.962961	1.232955	0.929877	0.960521
Mean	1.104481	0.835440	0.968207	1.168419	0.887388	0.964371	1.363654	1.054333	0.948889
SD	0.069298	0.054024	0.003968	0.086322	0.060196	0.005284	0.347218	0.325948	0.029362

Within the LSTM family, QLSTM achieves the best RMSE and R^2 in seven seeds, while the exact control leads in three. For MAE, however, the exact control leads in six seeds. The small, metric-dependent reversals confirm that the two models occupy nearly the same predictive regime. The compact LSTM remains inferior in every seed and metric, reinforcing the architecture-level advantage of the 16-state gate-specific formulation.

Table 9 Complete seed-wise LSTM-family test results. Lower RMSE and MAE are better; higher R^2 is better.

	QLSTM			Exact control			Compact LSTM		
Seed	**RMSE**	**MAE**	R^2	**RMSE**	**MAE**	R^2	**RMSE**	**MAE**	R^2
0	1.040878	0.756542	0.971863	1.012142	0.744765	0.973395	1.362021	1.034659	0.951823
1	1.006200	0.744642	0.973707	1.032715	0.780927	0.972303	1.519244	1.121937	0.940058
2	1.175457	0.908553	0.964117	1.095064	0.825798	0.968858	1.321382	0.997046	0.954655
3	1.053955	0.803821	0.971152	1.084460	0.785252	0.969458	1.356616	1.024574	0.952204
4	0.992658	0.760868	0.974410	1.081090	0.790178	0.969647	1.343013	1.012089	0.953158
5	1.005704	0.754940	0.973733	1.107480	0.824571	0.968147	1.394311	1.039945	0.949512
6	1.058952	0.796711	0.970878	1.067996	0.782142	0.970378	1.271591	0.963924	0.958008
7	1.083191	0.827220	0.969529	1.026739	0.761364	0.972623	1.436481	1.061478	0.946411
8	1.057794	0.803407	0.970941	1.064747	0.793528	0.970558	1.403198	1.019072	0.948866
9	1.001346	0.749635	0.973960	1.030789	0.752678	0.972406	1.523397	1.167773	0.939730
Mean	1.047613	0.790634	0.971429	1.060322	0.784120	0.970777	1.393126	1.044250	0.949443
SD	0.054324	0.050261	0.003049	0.032671	0.027017	0.001797	0.081369	0.060034	0.005959

The complete seed-wise structure is therefore consistent with the aggregate interpretation. The QGRU VQC effect is directionally coherent but not universal, while the QLSTM VQC effect is practically negligible relative to its matched classical replacement.

8.4 QGRU-family training and validation convergence

The convergence analysis examines whether the QGRU test advantage is reflected during model selection rather than arising solely from the held-out partition. Figure 7 shows the mean training and validation trajectories for QGRU, its exact gate-core control, and the compact GRU. All three models reduce their losses rapidly during the initial training phase, but their validation curves settle into clearly different regimes.

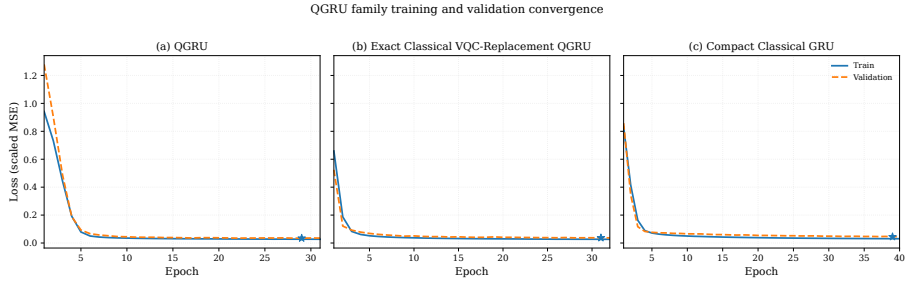


Fig. 7 Mean training and validation convergence for QGRU, its exact classical VQC-replacement control, and the compact classical GRU. Stars identify the minimum mean validation loss. Curves are displayed only while at least five of ten seeds contribute.

QGRU attains the lowest mean best validation MSE in the family, 0.03175 ± 0.00294, compared with 0.03579 ± 0.00391 for the exact control and 0.04866 ± 0.00974 for the compact GRU. The paired QGRU/control difference in best validation loss is 0.00404 in favour of QGRU ($p_t = 0.0131$, two-sided Wilcoxon $p = 0.0137$). Although this optimization comparison is a secondary analysis, it independently reproduces the ordering

observed for held-out RMSE. The agreement between validation-based checkpoint quality and test performance reduces the likelihood that the QGRU advantage is caused only by an isolated favourable test-set fluctuation.

The compact GRU reaches a relative plateau quickly, but its terminal validation level is substantially worse. Its broader seed dispersion and larger train–validation separation show that rapid reduction of the available validation gap does not imply superior absolute performance. The exact control converges to a markedly stronger regime than the compact GRU, demonstrating that the gate-specific pre- and post-projections provide substantial modelling capacity independently of the VQC. QGRU then improves further upon this already stronger architectural control.

8.5 QLSTM-family training and validation convergence

Figure 8 reveals that the LSTM family follows a different pattern. QLSTM and its exact control converge to closely overlapping validation regimes, whereas the compact LSTM remains clearly separated at a higher loss level. The exact control achieves a slightly lower mean best validation MSE (0.03249 ± 0.00098) than QLSTM (0.03405 ± 0.00326), but the paired difference is not statistically reliable ($p_t = 0.1408$, two-sided Wilcoxon $p = 0.1602$). The exact control also exhibits substantially lower seed dispersion.

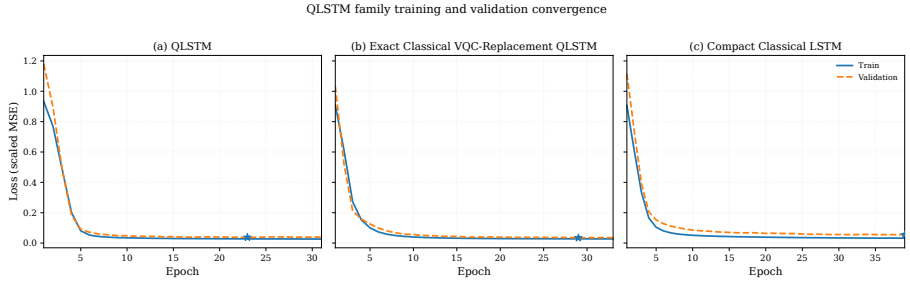


Fig. 8 Mean training and validation convergence for QLSTM, its exact classical VQC-replacement control, and the compact classical LSTM. QLSTM and its control converge to closely matching validation regimes, whereas the compact LSTM remains substantially higher.

This convergence pattern is consistent with the held-out test comparison. Replacing the four QLSTM VQCs with four parameter-matched classical cores does not materially degrade forecasting performance or checkpoint quality. At the same time, both 16-state models strongly outperform the compact LSTM. The figure therefore separates the benefit of the broader gate-specific architecture from the contribution of the quantum component: the architecture is effective, but the QLSTM VQC is not required to obtain that benefit in the present experiment.

8.6 Cross-architecture synthesis

The complete empirical evidence supports three nested conclusions. First, under the tested parameter budgets, both hybrid recurrent designs are substantially more parameter efficient than the compact vanilla baselines. This conclusion is supported by held-out RMSE, MAE, and R^2, as well as by the lower validation-loss regimes reached by the four 16-state architectures.

Second, exact gate-core replacement reveals different VQC contributions across the two recurrent families. QGRU retains a coherent advantage over its exact control in test RMSE, test MAE, best validation loss, and final generalization gap. Its paired test evidence is moderate rather than definitive, but the direction is consistent across predictive and optimization outcomes. QLSTM does not retain an equivalent advantage: its exact control is a predictive peer, attains a slightly lower mean best validation loss, and exhibits greater validation and generalization stability.

Third, both quantum models incur substantially higher simulator cost. The additional computational burden is particularly pronounced for QLSTM, which evaluates four VQC modules at every recurrent step but does not show a separately identifiable VQC-specific predictive benefit.

The central result is therefore not a universal quantum improvement, but a localized, architecture-dependent empirical effect. Under the present data and optimization regime, the tested four-qubit feature map is useful within the three-gate GRU recurrence, whereas the four-gate LSTM recurrence obtains essentially the same predictive benefit from an exactly matched classical nonlinear core.

8.7 Cross-model training and validation trajectories

The family-specific curves establish within-family attribution, while Figs. 9 and 10 place all six models on common axes. This broader comparison distinguishes failure to optimize from failure to generalize.

Figure 9 shows that the compact models continue to reduce their training losses throughout optimization. Their inferior test performance is therefore not explained by an inability to fit the training partition. The distinction is especially clear for the compact LSTM, whose training loss continues to decrease while validation improvement becomes limited.

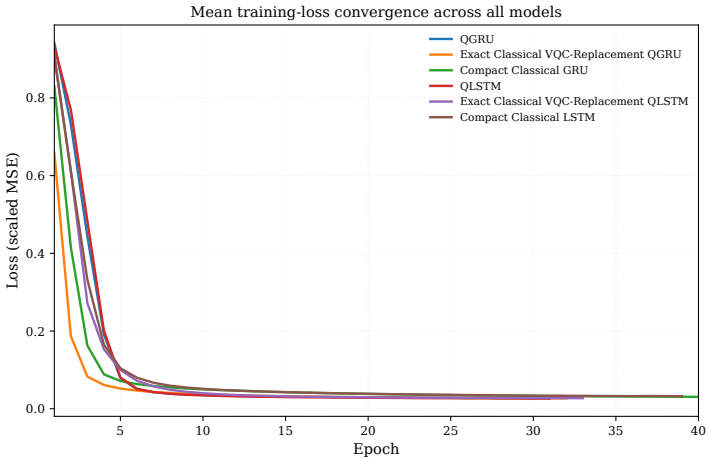


Fig. 9 Mean training-loss convergence for all six models. The plot is used to distinguish optimization progress from final generalization quality.

Figure 10 provides the corresponding generalization view. After the initial transient, the four 16-state architectures occupy a substantially lower and narrower validation-loss band than the compact baselines. The separation therefore arises not from a lack of training progress, but from the validation regimes reached under the tested architectural constraints. Within the lower-loss band, QGRU remains distinguishable from its control, whereas QLSTM and its control remain closely aligned.

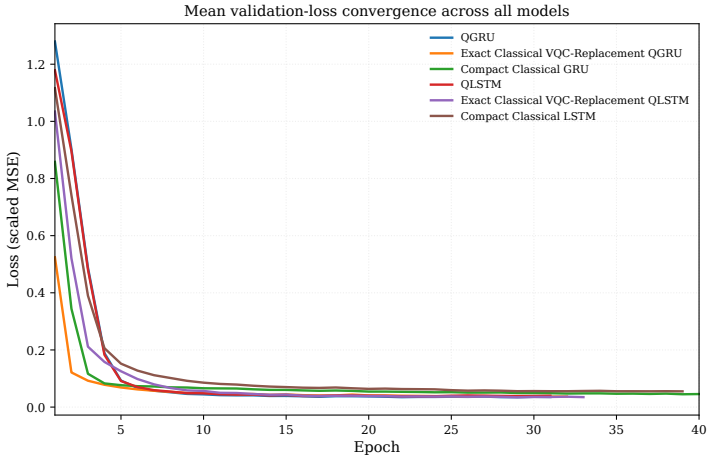


Fig. 10 Mean validation-loss convergence for all six models. The four 16-state quantum and exact-control architectures converge to lower validation regimes than the compact recurrent baselines.

8.8 Normalized convergence and convergence efficiency

Absolute validation loss determines checkpoint quality, but it does not by itself reveal how quickly each architecture approaches its own best attainable value. Equation (36) removes this scale dependence by expressing the fraction of the initial-to-best mean validation loss that remains at each epoch.

Figure 11 shows that all six models eliminate most of their achievable validation gap within approximately ten epochs. This finding must be interpreted together with the absolute curves. In particular, the compact GRU appears fast in normalized coordinates because it approaches a comparatively poor plateau rapidly. Normalized convergence therefore measures relative optimization efficiency rather than final predictive quality.

The seed-level e_{95} distributions in Fig. 12 complement the mean normalized trajectories. QGRU reaches 95% of its seed-specific validation improvement after 5.5 ± 0.5 epochs, compared with 9.0 ± 5.8 for its exact control. The paired difference is suggestive ($p_t = 0.0725$) and is driven primarily by a long tail in the control rather than by a uniform shift across all seeds.

QLSTM reaches the same threshold after 5.4 ± 0.5 epochs, compared with 7.3 ± 0.8 for its exact control and 8.1 ± 2.0 for the compact LSTM. The QLSTM/control difference is strong ($p_t = 7.37 \times 10^{-5}$; two-sided Wilcoxon $p = 0.00195$). However, the faster relative convergence of QLSTM does not produce a corresponding advantage in final

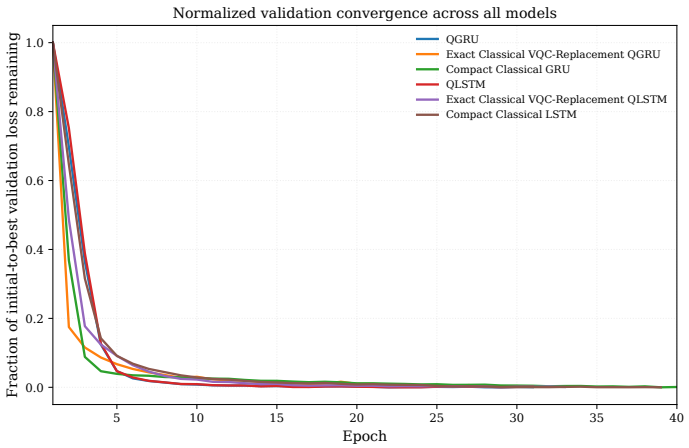


Fig. 11 Normalized validation convergence across all six models. The vertical axis is the fraction of the initial-to-best mean validation loss remaining.

test accuracy or best validation loss. The result demonstrates that optimization speed and final representational benefit are distinct properties.

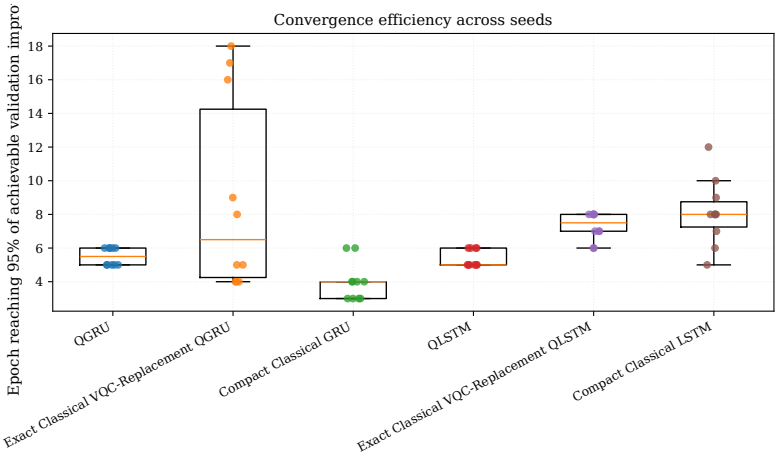


Fig. 12 Seed-wise epoch at which each model reaches 95% of its achievable validation-loss improvement. A low value indicates rapid approach to the seed's eventual best checkpoint, not necessarily a lower absolute validation loss.

8.9 Best-checkpoint quality, timing, and seed robustness

Table 10 jointly reports the quality of the selected checkpoints, the epochs at which they occur, the relative-convergence statistic e_{95}, and the total number of completed epochs. This separation is important because a model may approach its eventual regime quickly while continuing to achieve small checkpoint improvements later in training.

QGRU obtains the lowest mean best validation loss among the six models, whereas the exact QLSTM control has the smallest standard deviation. Across all models, the

Table 10 Checkpoint quality and optimization timing over ten random seeds. Values are reported as mean ± sample standard deviation.

Model	Best Validation MSE	Best Epoch	e_{95}	Epochs Completed
QGRU	0.03175 ± 0.00294	23.7 ± 6.6	5.5 ± 0.5	31.5 ± 6.3
Exact QGRU control	0.03579 ± 0.00391	23.7 ± 7.1	9.0 ± 5.8	31.4 ± 6.7
Compact GRU	0.04866 ± 0.00974	30.5 ± 14.5	4.0 ± 1.2	34.1 ± 11.9
QLSTM	0.03405 ± 0.00326	22.4 ± 6.9	5.4 ± 0.5	30.0 ± 6.1
Exact QLSTM control	0.03249 ± 0.00098	25.5 ± 5.9	7.3 ± 0.8	32.9 ± 4.9
Compact LSTM	0.05601 ± 0.01006	28.2 ± 7.2	8.1 ± 2.0	35.0 ± 5.9

mean best-checkpoint epoch lies between approximately 22 and 31, substantially later than the corresponding e_{95} values. Most of the available predictive structure is therefore learned during the initial training phase, while later epochs provide incremental refinement and determine the final selected checkpoint.

Figure 13 resolves the seed-wise variation underlying the table averages. QGRU has the lowest mean within the GRU family, while QLSTM and its control overlap substantially. The particularly narrow exact-QLSTM-control distribution confirms that its competitive mean performance is accompanied by strong robustness rather than being driven by a small number of favourable runs.

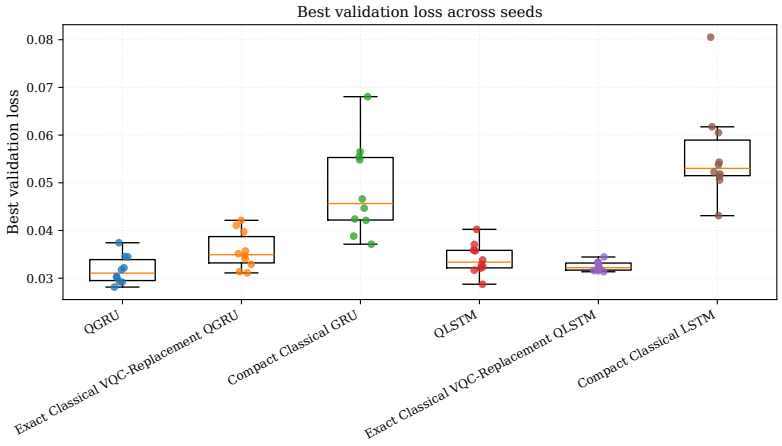


Fig. 13 Distribution of the best validation loss over ten seeds. QGRU has the lowest mean within the GRU family; QLSTM and its exact control overlap, with the control exhibiting the narrowest dispersion.

Figure 14 shows that checkpoint timing is more variable than checkpoint quality for several models. The broad compact-GRU distribution is particularly notable and indicates that its optimization trajectory depends strongly on initialization. The figure also shows that the quantum models do not systematically receive later or more favourable checkpoint selections than their exact controls.

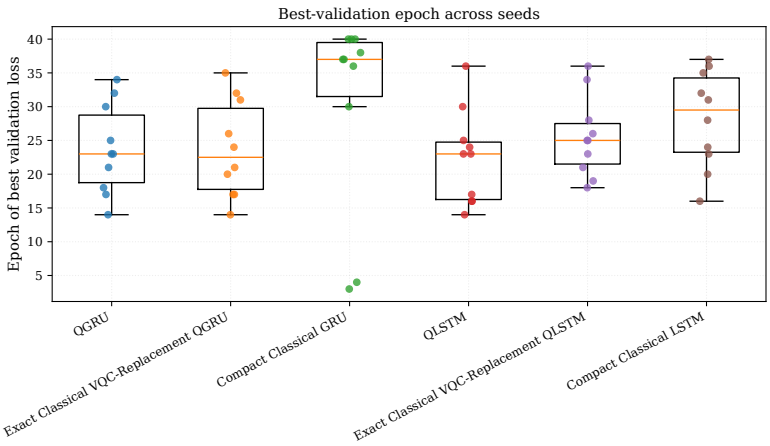


Fig. 14 Distribution of the epoch selected by minimum validation loss. The broad compact-GRU distribution reflects strong seed dependence in checkpoint timing.

The seed-wise heatmap in Fig. 15 exposes model seed interactions that are obscured by aggregate means. The compact baselines are not uniformly poor; rather, they combine higher average validation error with greater initialization sensitivity. QGRU shows a comparatively homogeneous seed profile, while the exact QLSTM control is the most uniform of all six models. The principal family-level ordering is therefore not produced by a single isolated seed.



Fig. 15 Seed-wise best validation MSE for every model. The heatmap exposes model–seed interactions that are hidden by aggregate means.

8.10 Early stopping and completed-epoch behaviour

Figure 16 reports the number of epochs completed before early stopping or the 40-epoch limit. The mean numbers of completed epochs are 31.5 for QGRU, 31.4 for its

exact control, 30.0 for QLSTM, 32.9 for its.5 for QGRU, 31.4 for its exact control exact control, 34.1 for the compact GRU, and 35.0 for the compact LSTM.

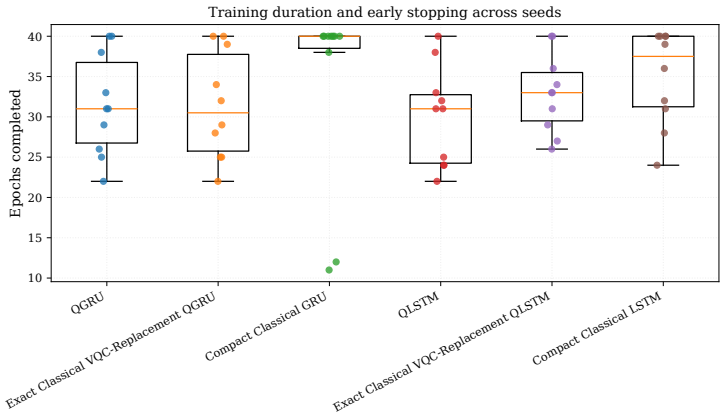


Fig. 16 Completed epochs before early stopping or the 40-epoch limit. Similar stopping distributions for the quantum models and exact controls reduce concern that unequal training duration caused the paired predictive results.

QGRU and its exact control are nearly identical in completed training duration, whereas the QLSTM control runs approximately 2.9 epochs longer on average than QLSTM. Neither paired difference is statistically compelling. The held-out performance ordering therefore cannot be explained by one member of a quantum/control pair receiving systematically more optimization steps.

The compact recurrent models tend to train for longer despite reaching worse best validation losses. Their disadvantage is thus not attributable to premature stopping. Instead, they continue optimizing toward validation plateaus that remain inferior under the selected parameter budget. The stopping analysis consequently reinforces the interpretation that the observed performance differences are architectural rather than procedural.

8.11 Generalization-gap evolution and final-gap robustness

Figure 17 tracks the mean generalization gap, $G_{m,e} = \overline{\mathcal{L}}^{\mathrm{val}}_{m,e} - \overline{\mathcal{L}}^{\mathrm{tr}}_{m,e}$. The curves exhibit large transients during the initial phase, when the models rapidly adapt to the training partition. After approximately five epochs, the gaps stabilize near small positive values. Early negative values indicate temporary epochs for which the mean validation loss is below the mean training loss and should not be interpreted as persistent negative generalization error.

The family-level final-gap statistics reinforce the architecture-dependent attribution. QGRU has the smallest mean final gap in the GRU family (0.00865 ± 0.00257), compared with 0.01133 ± 0.00336 for the exact control and 0.01714 ± 0.00574 for the compact GRU. The paired QGRU/control difference is 0.00268 in favour of QGRU ($p_t = 0.0480$, two-sided Wilcoxon $p = 0.0488$). The QGRU advantage therefore appears not only in held-out prediction and best validation loss, but also in the final separation between validation and training loss.

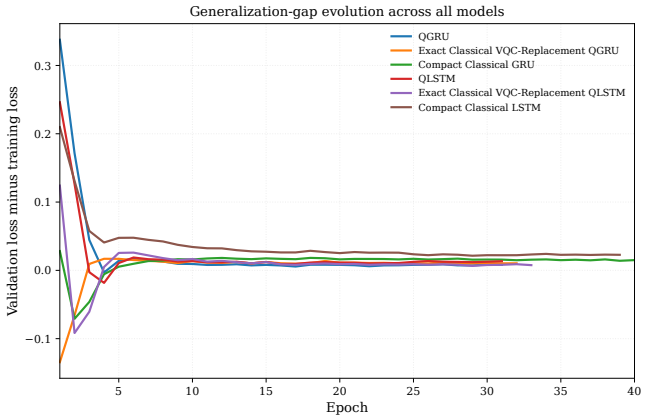


Fig. 17 Evolution of mean validation loss minus mean training loss. Early negative values indicate epochs for which validation loss temporarily falls below training loss during the initial transient.

In the LSTM family, the exact control has the smallest final gap (0.00808 ± 0.00180), followed by QLSTM (0.01122 ± 0.00302) and the compact LSTM (0.02569 ± 0.01193). The QLSTM/control difference favours the control ($p_t = 0.00923$, two-sided Wilcoxon $p = 0.00391$). Thus, the QLSTM VQC does not improve generalization-gap stability, despite reaching its relative convergence threshold more rapidly.

Table 11 Final generalization and logged runtime diagnostics. Runtime values are medians over seeds.

Model	**Final validation–training gap**	**Median seconds/epoch**	**Median total logged seconds**
QGRU	0.00865 ± 0.00257	17.30	606.60
Exact QGRU control	0.01133 ± 0.00336	0.50	15.30
Compact GRU	0.01714 ± 0.00574	$< 0.1^{\dagger}$	$< 0.1^{\dagger}$
QLSTM	0.01122 ± 0.00302	37.65	1,225.95
Exact QLSTM control	0.00808 ± 0.00180	0.40	13.20
Compact LSTM	0.02569 ± 0.01193	0.10	3.75

†The compact-GRU logger rounds epoch duration to one decimal place and records most epochs as 0.0 s; the true duration cannot be recovered from these logs.

Figure 18 shows the seed-wise distributions underlying these means. QGRU retains the smallest gap in the GRU family, while the exact classical control is the most stable member of the LSTM family. The opposite directions of the two exact-control comparisons provide additional evidence that the effect of the VQC cannot be generalized independently of the recurrent architecture in which it is embedded.

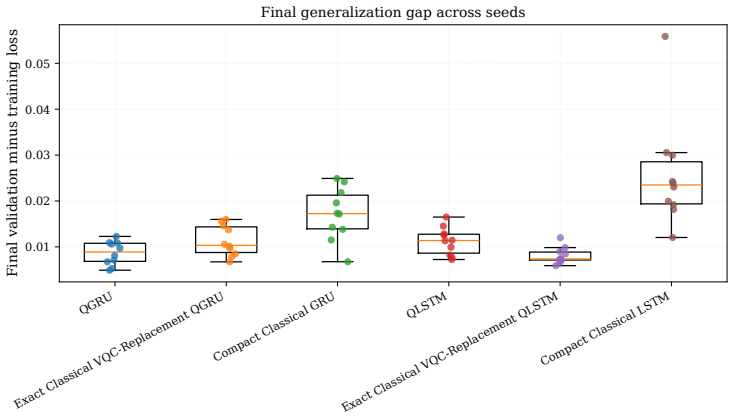


Fig. 18 Seed-wise distribution of the final validation–training gap. QGRU has the smallest gap in the GRU family, whereas the exact classical control has the smallest gap in the LSTM family.

8.12 Simulator runtime and total training cost

The predictive and optimization results must be interpreted together with the computational cost of statevector simulation. Figure 19 uses a logarithmic axis because the quantum and classical implementations differ by orders of magnitude. The median epoch time is 17.3 s for QGRU and 37.65 s for QLSTM, compared with 0.5 s and 0.4 s for their exact controls. In the observed software environment, replacing the VQC reduces median epoch time by approximately 35-fold in the GRU family and 94-fold in the LSTM family.

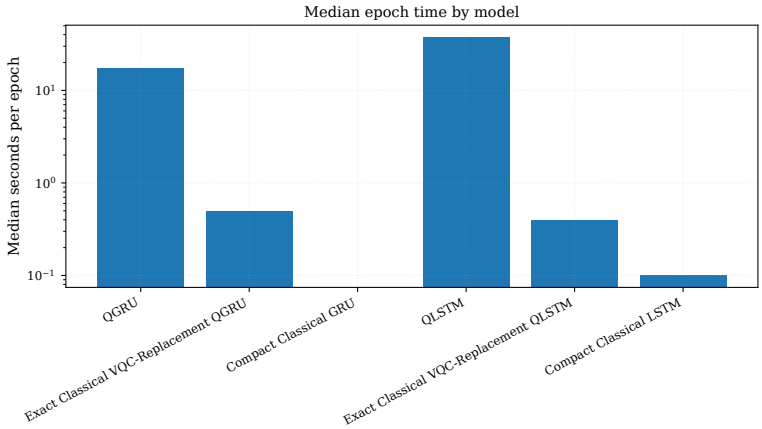


Fig. 19 Median logged seconds per epoch on a logarithmic scale. The compact GRU is not reliably represented because its timer rounds most epochs to 0.0 s.

The larger QLSTM cost is consistent with its recurrent structure: four independent VQC gates are evaluated at each recurrent step, compared with three for QGRU. The difference therefore reflects the number of circuit evaluations as well as the common simulator overhead.

Figure 20 reports the cumulative consequence of this per-epoch cost. Median total logged training times are 606.60 s for QGRU and 1,225.95 s for QLSTM, compared with 15.30 s and 13.20 s for their exact controls. These values quantify the practical cost of the present simulator-based experiments, but they are not hardware-normalized benchmarks. They include Python, simulator, data-loader, and system overheads, and the logs contain isolated interruptions. Median rather than mean timing is therefore reported.

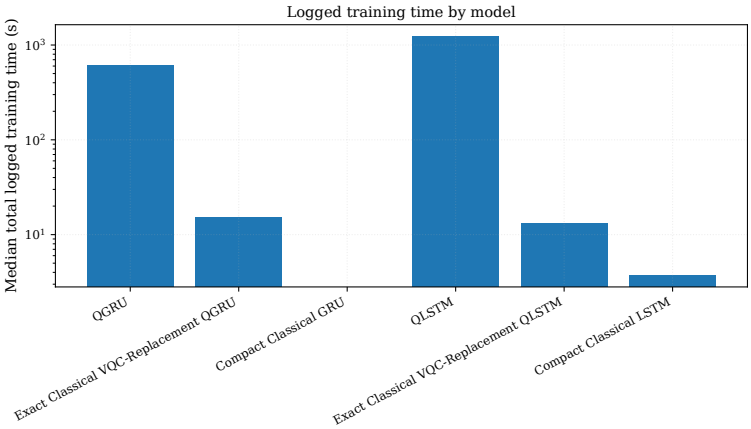


Fig. 20 Median total logged training time per seed on a logarithmic scale. These values measure the present classical-simulator implementation and are not quantum-hardware runtime claims.

The runtime results do not affect the predictive attribution analysis, but they make the accuracy–cost trade-off explicit. The QGRU VQC is associated with a moderate predictive advantage over its exact control at a substantially higher simulator cost. QLSTM incurs an even larger simulation cost without producing a separately identifiable predictive advantage over its classical gate-core surrogate.

8.13 Additional all-model convergence and stopping diagnostics

The remaining diagnostic figures consolidate the absolute optimization, stopping, generalization, runtime, and robustness patterns across the complete benchmark. Figures 21 and 22 show that the compact models begin with competitive initial losses but converge to inferior validation regimes. Their continued reduction in training loss confirms that the disadvantage arises from poorer validation behaviour rather than immediate optimization failure.

The best-checkpoint and completed-epoch distributions in Figs. 23 and 24 show that no model family receives a systematically larger optimization budget. The broad compact-GRU best-epoch distribution again demonstrates strong initialization dependence, while the completed-epoch distributions confirm that inferior compact-model performance is not caused by earlier termination.

Figure 25 confirms that the compact LSTM has the largest and most variable final validation-minus-training separation. QGRU and the exact QLSTM control exhibit the

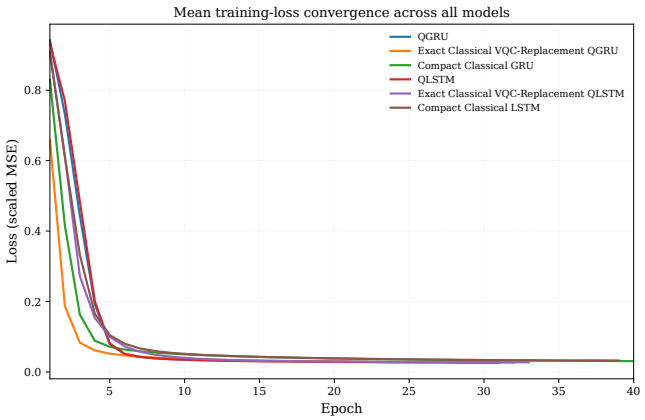


Fig. 21 Mean training-loss convergence for all six models.

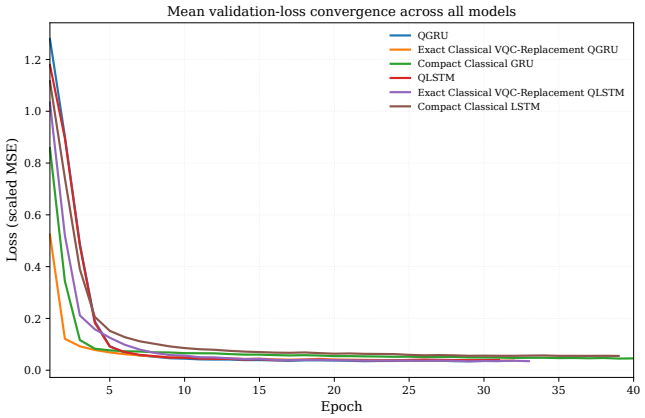


Fig. 22 Mean validation-loss convergence for all six models.

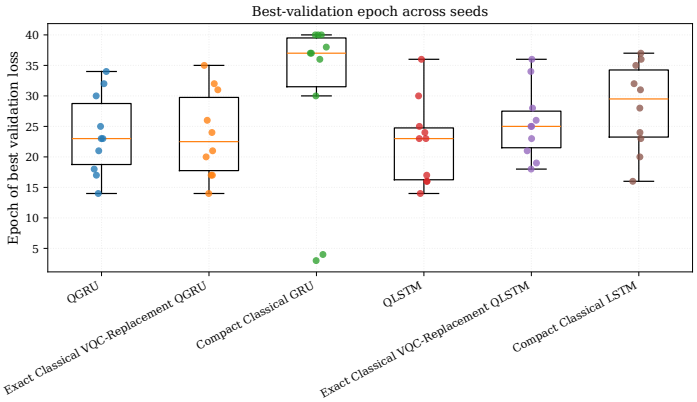


Fig. 23 Distribution of the epoch selected by minimum validation loss.

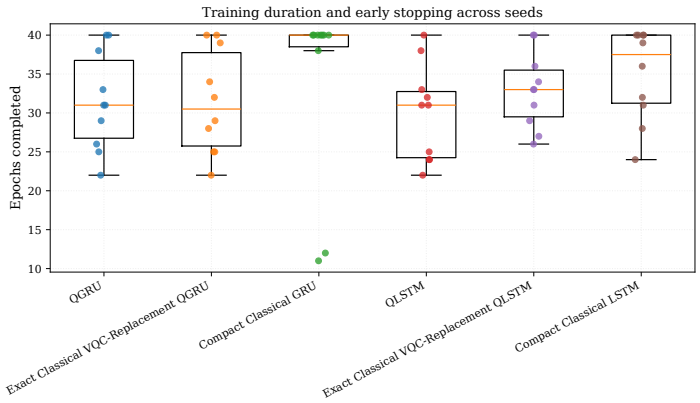


Fig. 24 Distribution of completed epochs before early stopping or the 40-epoch limit.

smallest gaps within their respective families, reproducing the opposing architecture-dependent pattern identified in the formal paired analysis. The total-time comparison in Fig. 26 makes the accuracy–cost trade-off explicit. The statevector-simulated quantum models require substantially longer training than the corresponding exact classical controls. These measurements characterize the present software environment and do not constitute hardware-normalized or quantum-speedup claims.

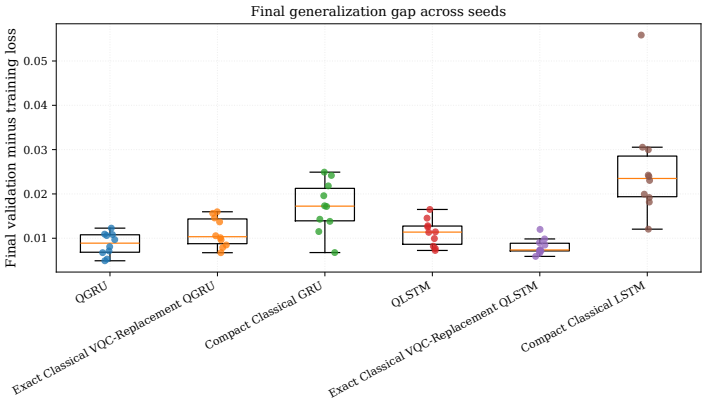


Fig. 25 Seed-wise distribution of the final validation-minus-training loss.

Finally, the all-model heatmap in Fig. 27 shows that the principal validation ordering is distributed across the ten seeds rather than being caused by a single initialization. It also highlights the high dispersion of the compact GRU and the uniformity of the exact QLSTM control.

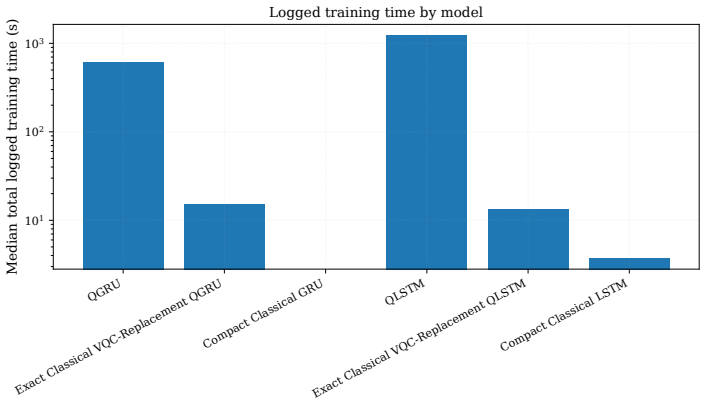


Fig. 26 Median total logged training time per seed on a logarithmic scale. These measurements describe the present software environment and are not hardware-normalized benchmarks.



Fig. 27 Seed-wise best-validation-loss heatmap across all six models.

8.14 Validation–test association for QLSTM and its exact control

The combined log-analysis script provides complete terminal test records for QLSTM and its exact classical control. Figure 28 compares their seed-wise test-RMSE distributions. The substantial overlap between the two distributions is consistent with the small aggregate RMSE difference and the non-significant paired tests.

Figure 29 relates the selected best validation loss to held-out test RMSE. The two models occupy overlapping validation–test regions, indicating that neither model defines a clearly superior checkpoint-quality regime. The plot supports the interpretation that their test-level differences are small relative to seed-wise variation.

Figure 30 compares completed training duration with held-out test RMSE. The overlap indicates that training for more epochs does not systematically yield better test

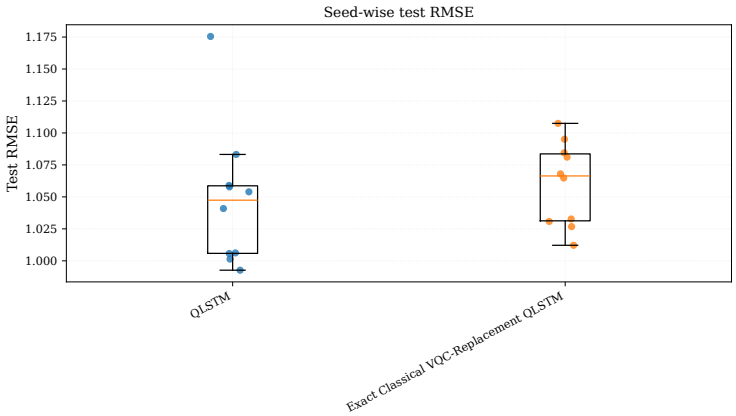


Fig. 28 Parsed seed-wise test-RMSE distribution for QLSTM and its exact control.

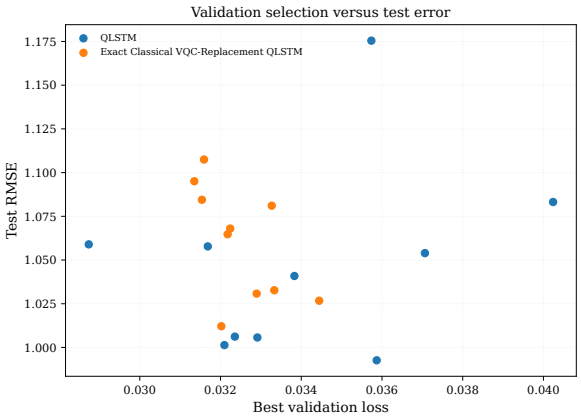


Fig. 29 Best validation loss versus held-out test RMSE for QLSTM and its exact control.

performance for either model. This observation is consistent with the earlier distinction between rapid relative convergence, later checkpoint refinement, and final predictive quality.

Because complete terminal records are available only for these two LSTM-family models, the three plots are not interpreted as an all-six-model correlation analysis. Within their intended scope, however, the overlapping RMSE, validation–test, and duration–performance distributions provide additional evidence that QLSTM and its exact classical control are practically equivalent under the reported experimental protocol.

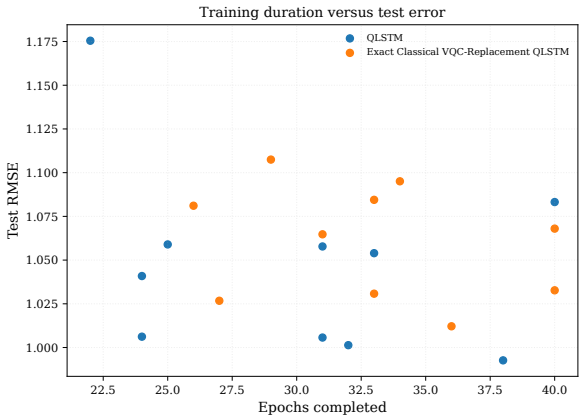


Fig. 30 Epochs completed versus held-out test RMSE for QLSTM and its exact control.

8.15 Duplicate-log numerical integrity audit

The seventh source log independently repeats the QGRU-versus-compact-GRU experiment. It is excluded from all primary aggregates to prevent double counting and is instead matched against the designated primary logs by model, seed, and epoch.

The audit identifies 656 common records. As shown in Table 12, the maximum absolute difference is 5×10^{-7} for both training and validation loss, while the median absolute differences are 2.5×10^{-7} and 2.4×10^{-7}, respectively. These discrepancies are consistent with differences in printed decimal precision rather than substantive disagreement between runs.

Table 12 Numerical integrity audit using the repeated QGRU/compact-GRU log.

Quantity	Matched records	Median absolute difference	Maximum absolute difference
Training loss	656	2.5×10^{-7}	5.0×10^{-7}
Validation loss	656	2.4×10^{-7}	5.0×10^{-7}

The agreement provides an end-to-end audit of model identification, seed alignment, epoch parsing, and loss extraction. It also confirms that excluding the duplicate log removes redundant records without discarding contradictory numerical evidence.

9 Discussion

9.1 Evidence chain for the QGRU-specific effect

The QGRU result is supported by several logically distinct and mutually reinforcing observations. In the held-out test comparison, QGRU achieves lower RMSE and MAE than the exact control in seven of ten matched seeds. The validation analysis independently identifies a lower best checkpoint loss and a smaller final generalization

gap. The convergence analysis further shows that QGRU approaches its near-optimal regime with lower seed-to-seed variation than the control's long-tailed e_{95} distribution. The agreement across held-out performance, validation quality, convergence behaviour, and generalization-gap diagnostics provides a coherent evidence chain for the QGRU-specific effect.

The controlled experimental design gives this result particular significance. The exact classical surrogate preserves the recurrent equations, hidden-state dimension, gate-specific projections, parameter count, output head, and optimization protocol. The observed difference therefore reflects the contribution of the VQC within an otherwise matched computational structure. This formulation directly tests whether the VQC contributes predictive value beyond that provided by the shared recurrent architecture.

The findings support an architecture-specific quantum contribution rather than a universal claim across all initializations and tasks. The ten-seed paired evaluation provides a meaningful basis for estimating the average effect, with QGRU leading in seven seeds and the exact control leading in three. This distribution demonstrates that the QGRU advantage is repeatable while also capturing the natural variability associated with stochastic optimization. Evaluations with additional seeds and datasets can further refine the confidence intervals and establish the range of conditions under which the effect persists.

The appropriate interpretation is therefore that QGRU exhibits a reproducible average benefit within the present controlled experiment. The strength of this conclusion arises from the consistency of the result across complementary predictive and optimization diagnostics rather than from reliance on a single aggregate metric.

9.2 Why QLSTM parity is scientifically informative

The practical parity between QLSTM and its exact control represents an equally informative scientific outcome. Both models outperform the compact LSTM by a large margin, demonstrating that the 16-state recurrent design with independent gate projections is highly effective under the selected parameter budget. The exact-control comparison then identifies the source of this improvement more precisely: a 36-parameter classical nonlinear core can reproduce the useful gate transformation without materially changing held-out performance.

This result demonstrates the value of separating architecture-level effectiveness from component-level attribution. A comparison restricted to QLSTM and the 1,018-parameter compact LSTM would reveal a substantial performance improvement. The additional comparison with the exact 1,025-parameter control shows that this improvement is mainly associated with the wider recurrent structure and its independent gate-specific projections.

Accordingly, QLSTM parity complements the QGRU finding and strengthens the overall contribution of the study. The experiments identify one recurrent setting in which the VQC provides a measurable average benefit and another in which an exactly matched classical surrogate provides equivalent predictive capability. Both outcomes are valuable for controlled quantum machine learning because they establish where a quantum component contributes additional predictive value and where comparable performance can be achieved through a classical transformation.

The QLSTM result also provides a practically useful design insight. When the recurrent architecture already contains four independent gate pathways, an additive cell state, and gate-specific pre- and post-projections, the resulting classical structure may provide sufficient nonlinear flexibility for one-step forecasting. In this setting, the exact surrogate offers a computationally efficient alternative while preserving the predictive benefit of the broader architecture.

9.3 Potential mechanisms behind architecture dependence

The contrasting QGRU and QLSTM outcomes indicate that the contribution of the VQC depends on how the transformed gate outputs interact with the recurrent state. GRU and LSTM differ substantially in this respect. In GRU, the update and reset signals directly regulate the interpolation between the previous hidden state and the candidate state. A bounded four-dimensional quantum feature map can therefore influence both the transition coefficient and the candidate-state geometry within a single recurrent-state pathway.

In LSTM, the forget, input, candidate, and output transformations interact with a separate additive cell state. The surrounding pre-/post-projections and the availability of four gate pathways may already provide sufficient flexibility for one-step forecasting, enabling the classical surrogate to reproduce the useful gate transformation. This interpretation is consistent with the observed QLSTM parity and with the strong performance of both 16-state LSTM-family models relative to the compact baseline.

These observations motivate an architecture-dependent interpretation of quantum recurrent modelling. The usefulness of a VQC may depend not only on its intrinsic expressibility but also on the recurrent pathway through which its outputs influence the hidden and memory states. The QGRU architecture provides a comparatively direct route from the quantum-transformed gate features to the recurrent-state transition, whereas the QLSTM architecture distributes its transformation across multiple interacting gates and an additive memory pathway.

This explanation remains a mechanistic hypothesis that can be examined through targeted analysis. Measurements of gate activations, Jacobian spectra, hidden-state geometry, circuit expressibility, and gradient statistics would clarify how the quantum and classical gate cores shape the recurrent dynamics. Repeating the exact-control intervention under different qubit counts, circuit depths, and entanglement patterns would further determine whether the observed architecture dependence arises from state-update geometry, optimization behaviour, representational redundancy, or interactions among these [10, 11].

9.4 Convergence speed versus checkpoint quality

The normalized convergence and e_{95} analyses establish that rapid progress toward a model's own optimum and the quality of that optimum are complementary properties. Compact GRU reaches 95% of its achievable improvement rapidly because its validation curve approaches its final plateau within relatively few epochs. The absolute validation analysis shows that this plateau remains higher than those reached by the 16-state

architectures. QLSTM reaches its near-best regime faster than its exact control, while the control obtains a slightly lower and more stable best validation loss.

These findings demonstrate the importance of reporting optimization speed, final checkpoint quality, and held-out generalization separately. The normalized statistic captures how efficiently a model approaches its own attainable regime, whereas the absolute validation loss measures the quality of that regime. The held-out metrics then determine whether the selected checkpoint transfers effectively to the final chronological test segment.

The best checkpoints occur substantially later than e_{95} for all models. This separation indicates a two-phase optimization process: rapid acquisition of the dominant temporal structure followed by slower and more incremental refinement. Early stopping provides an effective mechanism for selecting checkpoints during this second phase. The similar completed-epoch distributions of the quantum models and their exact controls further show that the paired outcomes arise under comparable optimization budgets.

The convergence analysis therefore strengthens the component-attribution results. The QGRU advantage is supported by checkpoint quality and generalization rather than by training duration alone. Similarly, QLSTM reaches its near-best regime rapidly, while its exact control achieves comparable final predictive performance. The two recurrent families consequently illustrate distinct relationships between optimization efficiency and representational benefit.

9.5 Validation selection and held-out metrics

Validation MSE and held-out RMSE provide complementary views of model performance. They are evaluated on different chronological segments, and MAE weights residuals differently from squared-error metrics. The exact QLSTM control achieves the lowest LSTM-family validation MSE and MAE, whereas QLSTM obtains marginally lower test RMSE and higher R^2. This small difference is consistent with chronological distribution variation and metric sensitivity.

The validation diagnostics are therefore used to explain optimization behaviour and checkpoint selection, while the pre-specified held-out metrics remain the primary predictive outcomes. This separation ensures that checkpoint-selection evidence and final forecasting evidence are interpreted according to their distinct roles. It also explains how closely matched models can exchange rank across metrics while remaining within the same practical predictive regime.

Within this framework, QGRU exhibits a consistent advantage across validation and held-out analyses. QLSTM and its exact control remain closely matched, with each model showing small advantages under different metrics. The overall interpretation is therefore based on the complete pattern across chronological partitions and complementary error measures rather than on the ranking produced by any single metric.

This multi-metric perspective is particularly valuable for time-series forecasting because different chronological segments may contain different levels of variability and different residual distributions. Reporting RMSE, MAE, R^2, validation loss, and generalization-gap behaviour together provides a more complete characterization of predictive performance and checkpoint robustness.

9.6 Accuracy cost trade-off

Statevector simulation represents the principal practical cost associated with the quantum models. The exact classical controls preserve the recurrent equations and parameter count while avoiding quantum-state simulation, resulting in substantial reductions in epoch and total training time. QLSTM is approximately twice as expensive as QGRU because it evaluates four independent VQCs at every time step instead of three.

The present QGRU result should therefore be interpreted as an inductive-bias and predictive-accuracy contribution. QGRU achieves a measurable average benefit over its exact control while requiring substantially greater simulator computation. QLSTM incurs an even larger simulation cost while providing predictive performance comparable to that of its exact classical surrogate. Together, these outcomes define a clear accuracy–cost trade-off for the two hybrid recurrent families.

This comparison supports a selective approach to quantum recurrent modelling. In QGRU, the additional simulation cost accompanies a coherent improvement across held-out error, validation quality, and final generalization gap. In QLSTM, the exact classical core provides a computationally efficient alternative while retaining the predictive quality of the broader architecture. The results therefore identify both a promising use case for the VQC and a setting in which classical replacement is practically advantageous.

Practical hardware relevance requires a broader cost model than statevector simulation alone. Finite-shot hardware execution introduces sampling variance, queueing, communication latency, device calibration drift, topology constraints, and potentially error mitigation. Future hardware comparisons should report circuit calls, shots, latency, energy or resource cost, failure handling, and predictive accuracy under equal wall-clock or monetary budgets.

Such evaluations would extend the present controlled benchmark from simulator-based component attribution to end-to-end deployment analysis. They would also establish whether the architecture-specific QGRU benefit remains observable under finite-shot noise and realistic execution constraints.

9.7 Implications for controlled quantum-machine-learning evaluation

The study motivates a structured hierarchy of controls for hybrid QML. First, a compact conventional baseline evaluates parameter-budget effectiveness. Second, an architecture-matched classical surrogate isolates the contribution of the quantum component. Third, optimization and generalization diagnostics determine whether final performance is associated with convergence behaviour, checkpoint stability, or train–validation separation. Fourth, runtime and numerical-integrity audits establish practical cost and numerical provenance.

Applying this hierarchy leads to a substantially more informative scientific conclusion. The compact-baseline comparisons show that both QGRU and QLSTM provide strong predictive performance under constrained parameter budgets. The exact-control intervention then reveals a more precise architecture-dependent result: the QGRU VQC

contributes a measurable average effect, whereas the QLSTM VQC can be replaced by a matched classical core while preserving predictive performance.

This distinction represents the principal methodological and empirical contribution of the study. Controlled quantum-machine-learning evaluation should establish both whether a hybrid model outperforms a conventional baseline and whether the quantum component remains predictively valuable after preserving the surrounding architecture, recurrent equations, parameter count, and optimization protocol.

Under this evaluation standard, the QGRU advantage and the QLSTM parity are equally meaningful outcomes. The QGRU result identifies a recurrent configuration in which the VQC adds predictive value beyond an exact classical replacement. The QLSTM result identifies a configuration in which the surrounding recurrent architecture captures the dominant benefit and an equivalent classical transformation is sufficient.

The findings therefore support a targeted and evidence-driven view of hybrid quantum recurrent modelling. Quantum components can be evaluated as architecture-dependent nonlinear modules whose contribution is established through exact interventions rather than inferred solely from broad baseline comparisons. This perspective promotes more precise model design, clearer component attribution, and more efficient allocation of quantum computational [19–21].

10 Conclusion

This work presents a controlled evaluation of variational quantum recurrent gates under two complementary classical comparison regimes, with predictive accuracy, convergence behaviour, checkpoint quality, generalization, runtime, and numerical integrity treated as integral empirical outcomes. Under approximately equal parameter budgets, QGRU and QLSTM substantially outperform the compact vanilla GRU and LSTM baselines. These results establish the effectiveness of the complete hybrid recurrent architectures and demonstrate the importance of evaluating parameter efficiency together with component-level attribution.

The exact gate-core interventions provide a direct assessment of the contribution made by the variational quantum circuits. Relative to the exact 773-parameter classical surrogate, QGRU reduces test RMSE by 5.47% and MAE by 5.85%, achieves better performance in seven of ten paired seeds, obtains a lower mean best validation MSE, and exhibits a smaller final generalization gap. The agreement across held-out prediction, checkpoint quality, convergence, and generalization diagnostics provides consistent evidence that the VQC contributes predictive value within the QGRU recurrence. The ten-seed paired evaluation captures this average effect across stochastic initializations, while broader replications can further characterize its magnitude across datasets, forecasting horizons, and optimization conditions.

The QLSTM comparison provides a complementary and equally informative result. QLSTM and its exact 1,025-parameter classical surrogate occupy the same practical predictive regime, while both substantially outperform the compact LSTM. This outcome shows that the wider recurrent state, independent gate-specific projections, and preserved LSTM memory pathway account for the dominant improvement in this

architecture. A parameter-matched classical nonlinear core is therefore able to retain the predictive capability of the complete gate-specific QLSTM structure.

Taken together, the two recurrent families demonstrate that the contribution of a variational quantum gate depends on the computational pathway through which its transformed features enter the recurrent state. In QGRU, the VQC is associated with consistent improvements across multiple predictive and optimization endpoints. In QLSTM, the surrounding recurrent architecture provides sufficient flexibility for the matched classical core to reproduce the observed performance. This distinction enables a more precise interpretation of hybrid quantum recurrent models than comparisons based only on compact conventional baselines.

The run-log analysis further shows that final test metrics alone do not fully characterize model behaviour. The quantum models reach their near-best validation regimes rapidly, while checkpoint refinement continues over subsequent epochs. Their exact classical controls require similar numbers of training epochs, indicating that the paired outcomes are obtained under comparable optimization durations. In the observed simulator environment, the exact controls execute approximately 35 times faster per epoch for GRU and 94 times faster for LSTM, quantifying the accuracy–cost trade-off associated with statevector simulation. The duplicate-log audit additionally confirms 656 repeated epoch records to within 5×10^{-7} loss units, supporting the numerical consistency and reproducibility of the reported analysis.

The findings highlight the value of combining compact parameter-matched baselines with exact architecture-preserving controls. The compact comparisons establish the effectiveness of the complete recurrent designs, whereas the exact interventions determine whether the quantum gate core contributes beyond the shared architecture. This evaluation framework provides a rigorous basis for identifying recurrent configurations in which variational quantum features add predictive value and configurations in which an equivalent classical transformation provides the same practical outcome.

11 Future Directions

Future work will extend the evaluation across additional random seeds, chronological origins, forecasting horizons, datasets, and stronger classical surrogate models. Ablation studies will also examine the effects of qubit count, circuit depth, data re-uploading, entanglement structure, gate activations, hidden-state dynamics, and gradient behaviour.

The models will first be tested on finite-shot noisy quantum simulators with hardware-inspired gate, readout, decoherence, and sampling noise. The most robust QGRU and QLSTM configurations will then be executed on real quantum hardware using available quantum processing units (QPUs). The hardware experiments will assess predictive accuracy under realistic device connectivity, transpilation overhead, calibration drift, queueing latency, finite-shot uncertainty, and error-mitigation requirements.

This simulator-to-hardware evaluation will determine whether the observed QGRU-specific benefit remains measurable under realistic noise and physical QPU execution while clarifying the associated accuracy–cost trade-off[25, 26].

Code availability

The source code, trained models, and supporting scripts required to reproduce the experiments and results reported in this study will be made available by the corresponding author upon reasonable request.

Funding

This research was conducted at the Centre for Development of Advanced Computing (C–DAC), Noida, with funding support from the Ministry of Electronics & Information Technology (MeitY), Government of India.

Statements

The authors Abhishek Tiwari, Bhushan Kape, Sachin Kumar, Pankaj Tyagi and Geeta Singh declare that they have no competing interests.

Declarations

Portions of this manuscript were refined using AI-assisted copy-editing tools solely to improve clarity, readability, and language. The scientific content, methodology, analysis, results, and conclusions are entirely the work of the authors.

Author contributions

CRediT Taxonomy– Conceptualization: Bhushan Kape, Abhishek Tiwari; Methodology: Bhushan Kape, Abhishek Tiwari; Software: Bhushan Kape, Abhishek Tiwari; Data Curation: Bhushan Kape, Abhishek Tiwari; Validation: Bhushan Kape, Abhishek Tiwari; Formal Analysis: Bhushan Kape, Abhishek Tiwari; Investigation: Bhushan Kape, Abhishek Tiwari; Visualization: Bhushan Kape, Abhishek Tiwari; Writing–Original Draft Preparation: Bhushan Kape, Abhishek Tiwari; Writing–Review & Editing: Bhushan Kape, Abhishek Tiwari; Supervision: Abhishek Tiwari, Sachin Kumar, Pankaj Tyagi, Geeta Singh

References

[1] Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Computation* **9**(8), 1735–1780 (1997). https://doi.org/10.1162/neco.1997.9.8.1735

[2] Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1724–1734 (2014). https://doi.org/10.3115/v1/D14-1179

[3] Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., Woo, W.-C.: Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In: *Advances in Neural Information Processing Systems*, vol. 28, pp. 802–810 (2015)

[4] Karevan, Z., Suykens, J.A.K.: Transductive LSTM for time-series prediction: An application to weather forecasting. *Neural Networks* **125**, 1–9 (2020). https://doi.org/10.1016/j.neunet.2019.12.030

[5] Hersbach, H., Bell, B., Berrisford, P., et al.: The ERA5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society* **146**(730), 1999–2049 (2020). https://doi.org/10.1002/qj.3803

[6] Mitarai, K., Negoro, M., Kitagawa, M., Fujii, K.: Quantum circuit learning. *Physical Review A* **98**, 032309 (2018). https://doi.org/10.1103/PhysRevA.98.032309

[7] Pérez-Salinas, A., Cervera-Lierta, A., Gil-Fuster, E., Latorre, J.I.: Data re-uploading for a universal quantum classifier. *Quantum* **4**, 226 (2020). https://doi.org/10.22331/q-2020-02-06-226

[8] Cerezo, M., Arrasmith, A., Babbush, R., et al.: Variational quantum algorithms. *Nature Reviews Physics* **3**, 625–644 (2021). https://doi.org/10.1038/s42254-021-00348-9

[9] Bergholm, V., Izaac, J., Schuld, M., Gogolin, C., Blank, C., McKiernan, K., Killoran, N.: PennyLane: Automatic differentiation of hybrid quantum–classical computations. arXiv:1811.04968 (2018). https://arxiv.org/abs/1811.04968

[10] Sim, S., Johnson, P.D., Aspuru-Guzik, A.: Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum–classical algorithms. *Advanced Quantum Technologies* **2**(12), 1900070 (2019). https://doi.org/10.1002/qute.201900070

[11] McClean, J.R., Boixo, S., Smelyanskiy, V.N., Babbush, R., Neven, H.: Barren plateaus in quantum neural network training landscapes. *Nature Communications* **9**, 4812 (2018). https://doi.org/10.1038/s41467-018-07090-4

[12] Bausch, J.: Recurrent quantum neural networks. arXiv:2006.14619 (2020). https://arxiv.org/abs/2006.14619

[13] Chen, S.Y.-C., Yoo, S., Fang, Y.-L.L.: Quantum long short-term memory. In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 8622–8626 (2022). https://doi.org/10.1109/ICASSP43922.2022.9747369

[14] Li, Y., Wang, Z., Han, R., Shi, S., Li, J., Shang, R., Zheng, H., Zhong, G., Gu, Y.: Quantum recurrent neural networks for sequential learning. *Neural Networks*

166, 148–161 (2023). https://doi.org/10.1016/j.neunet.2023.07.003

[15] Ceschini, A., Rosato, A., Panella, M.: A variational approach to quantum gated recurrent units. *Journal of Physics Communications* **8**(8), 085004 (2024). https://doi.org/10.1088/2399-6528/ad6db7

[16] Li, Y., Wang, Z., Xing, R., Shao, C., Shi, S., Li, J., Zhong, G., Gu, Y.: Quantum gated recurrent neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **47**(4), 2493–2504 (2025). https://doi.org/10.1109/TPAMI.2024.3519605

[17] Sikora, H., Shridevi, S., Groppe, S.: Weather forecasting using quantum-based LSTM: A comparative analysis. *Results in Engineering* **28**, 107995 (2025). https://doi.org/10.1016/j.rineng.2025.107995

[18] Moon, K.-H., Jeong, S.-G., Hwang, W.-J.: QSegRNN: Quantum segment recurrent neural network for time series forecasting. *EPJ Quantum Technology* **12**, 32 (2025). https://doi.org/10.1140/epjqt/s40507-025-00333-6

[19] Huang, H.-Y., Broughton, M., Mohseni, M., Babbush, R., Boixo, S., Neven, H., McClean, J.R.: Power of data in quantum machine learning. *Nature Communications* **12**, 2631 (2021). https://doi.org/10.1038/s41467-021-22539-9

[20] Bowles, J., Ahmed, S., Schuld, M.: Better than classical? The subtle art of benchmarking quantum machine learning models. arXiv:2403.07059 (2024). https://arxiv.org/abs/2403.07059

[21] Jones, C., Kraus, N., Bhardwaj, P., Adler, M., Schrödl-Baumann, M., Zambrano Manrique, D.: Benchmarking quantum models for time-series forecasting. arXiv:2412.13878 (2024). https://arxiv.org/abs/2412.13878

[22] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: *Proceedings of the International Conference on Learning Representations* (2015). arXiv:1412.6980. https://arxiv.org/abs/1412.6980

[23] Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv:1607.06450 (2016). https://arxiv.org/abs/1607.06450

[24] Paszke, A., Gross, S., Massa, F., et al.: PyTorch: An imperative style, high-performance deep learning library. In: *Advances in Neural Information Processing Systems*, vol. 32, pp. 8024–8035 (2019). https://arxiv.org/abs/1912.01703

[25] Preskill, J.: Quantum computing in the NISQ era and beyond. *Quantum* **2**, 79 (2018). https://doi.org/10.22331/q-2018-08-06-79

[26] Temme, K., Bravyi, S., Gambetta, J.M.: Error mitigation for short-depth quantum circuits. *Physical Review Letters* **119**, 180509 (2017).

https://doi.org/10.1103/PhysRevLett.119.180509