Deep Learning for forecasting Quantum Dynamics

Emanuele Colecchia
Matricola: 276527

Pierluigi Trocini
Matricola: 280977

January 30, 2026

Abstract

Modeling multi-qubit quantum dynamics is challenging due to the exponential complexity of entanglement. This study presents a Deep Learning framework using **LSTM** and **Transformer** architectures to forecast quantum trajectories and bypass classical simulation limits. By optimizing input time windows and integrating a **Super-Resolution** model, we enhance the granularity of predicted states and preserve high-frequency oscillations. Furthermore, **attention maps** are analyzed to provide physical explainability. Our results demonstrate that this hybrid approach accurately reconstructs complex quantum dynamics, offering a scalable tool for analyzing next-generation quantum hardware.

1 Introduction

1.1 Context and Motivation

Quantum mechanics is the cornerstone of modern technology, enabling innovations from transistors to Google's **Willow** quantum processor. The field's significance is highlighted by recent **Nobel Prizes** in both Physics (2022, 2023, 2025) and Machine Learning (2024), marking a pivotal intersection between the physical sciences and AI. Despite these advances, the exponential complexity driven by **entanglement** renders large-scale quantum chips (approximately 100+ qubits) a computational **"Black Box"**. Classical simulations fail to track these dynamics efficiently. This project aims to leverage **Deep Learning** to model and forecast these complex behaviors, bypassing classical limits to facilitate the design of next-generation quantum hardware.

2 Technological background

2.1 Deep Learning Models and Architectures

2.1.1 Long-Short Term Memory (LSTM) model

Long Short-Term Memory (LSTM) networks are designed to process sequences recurrently by maintaining a hidden state h_t and a cell state c_t:

$$h_t, c_t = \mathrm{LSTM}(x_t, h_{t-1}, c_{t-1}) \tag{1}$$

The architecture utilizes three internal gates (input, forget, and output) to regulate the flow of information. This structure allows the model to learn long-term dependencies in quantum trajectories while mitigating the vanishing gradient problem common in standard recurrent networks.

2.2 Transformer Architecture

2.2.1 Core Components

The Transformer architecture bypasses recurrence in favor of self-attention mechanisms, enabling parallel processing of quantum data and improved long-range dependency capture.

Time Embedding and Positional Encoding

To process the 55 physical features, we employ a `TimeSeriesEmbedding` layer. Since the attention mechanism is permutation-invariant, positional information is injected using sinusoidal functions to maintain the temporal order of the quantum states:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d}}\right), \quad PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d}}\right) \tag{2}$$

Self-Attention Mechanism

The core innovation of this model is the ability for each position to attend to all other positions simultaneously via Query (Q), Key (K), and Value (V) projections:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{3}$$

This mechanism allows the network to identify which past states are most relevant to future dynamics, providing a direct way to explain predictions.

Multi-Head Attention

By using h parallel attention heads (optimized to 2 or 4 through tuning), the model can simultaneously track different temporal scales and patterns within the quantum system:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(head_1, ..., head_h)W^O \tag{4}$$

3 Experiment

3.1 Research pipeline

The research methodology followed a five-step structured pipeline:

1. **Data Preprocessing**: Trajectories of a dynamical quantum system were normalized and restructured into input-target pairs using a sliding window technique.
2. **Time windows selec**: Preliminary **LSTM** and **Transformer** "quick models" were tested to identify the minimum input window required for accurate long-term forecasting.
3. **Hyperparameter Tuning**: We utilized the `keras-tuner` library to automate the optimization of architectural variables, including embedding dimensions, attention heads, and dropout rates.
4. **Model Implementation**: Final LSTM and Transformer architectures were constructed based on optimized parameters to capture recurrent and non-local temporal dependencies, respectively.
5. **Evaluation and Explainability**: Model accuracy was assessed via MSE and MAE. For the Transformer, **Attention Maps** were analyzed to ensure the model's logic aligned with physical principles like temporal locality and causality.

To preserve vital high-frequency quantum oscillations, a **Super-Resolution** module was integrated. This component reconstructs high-density sequences (100 timesteps) from low-resolution inputs (50 timesteps), ensuring high-fidelity predictions.

4 Dataset & Preprocessing

4.1 Dataset Description

The experimental data consist of 400 independent trajectories generated from a simulated dynamical quantum system. Each trajectory represents a continuous evolution of the system's state over time, characterized by the following parameters:

- **Trajectories**: 400 distinct temporal sequences providing a diverse set of initial conditions and dynamical evolutions.
- **Temporal Resolution**: Each trajectory contains 1,001 equidistant timesteps, capturing both low-frequency trends and high-frequency quantum oscillations.
- **Feature Space**: The raw data include 56 dimensions, consisting of 55 physical state variables and a single temporal marker.
- **Physical Domain**: The system reflects continuous dynamics, where the state variables represent the evolution of quantum properties subject to entanglement and non-classical interactions.

During the initial preprocessing phase, the time column was decoupled from the feature matrix, as the temporal information is implicitly handled by the sequence-based architectures (LSTM and Transformer). This results in a final input dimensionality of 55 features per timestep.

Table 1: Preview of the quantum trajectories dataset (first 5 samples)

Index	**0 (Time)**	**1**	**2**	**3**	**4**	...	**52**	**53**	**54**	**55**
0	0.00	-0.0015	-0.0509	-0.0041	0.0010	...	0.0168	-0.0195	0.0021	-0.0148
1	0.02	-0.0018	-0.0510	-0.0038	0.0009	...	0.0169	-0.0191	0.0023	-0.0138
2	0.04	-0.0022	-0.0513	-0.0037	0.0006	...	0.0171	-0.0183	0.0027	-0.0123
3	0.06	-0.0028	-0.0518	-0.0038	0.0001	...	0.0173	-0.0171	0.0033	-0.0105
4	0.08	-0.0035	-0.0524	-0.0041	-0.0005	...	0.0174	-0.0155	0.0040	-0.0082

Dataset dimensions: 400,400 rows × 56 columns

4.2 Preprocessing Pipeline

4.2.1 Data Normalization

To ensure numerical stability and facilitate faster convergence during the training process, all features were normalized within the range $[-1, 1]$. This was achieved using the `MinMaxScaler` implementation from the `scikit-learn` Python library, according to the following transformation:

$$x_{norm} = 2 \cdot \frac{x - x_{min}}{x_{max} - x_{min}} - 1 \tag{5}$$

This scaling procedure assigns equal weight to all 55 physical variables, preventing features with larger absolute magnitudes from dominating the gradient updates. Furthermore, mapping the data to this specific interval ensures full compatibility with the *tanh* activation functions employed in the neural architectures, as it aligns the input distribution with the most responsive regions of the non-linear mapping.

5 Time Windows Selection

The identification of an optimal input window is a critical step in balancing predictive performance with computational efficiency. The candidate windows were strategically selected to ensure perfect alignment with the discrete timesteps of the dataset, while simultaneously considering the hardware constraints and memory limits available for the training process. Specifically, five candidate windows were evaluated, representing increasing scales of historical context. These windows, expressed in both physical time and their corresponding number of timesteps, are:

Table 2: Candidate input windows and their corresponding physical time durations.

Window Size (Timesteps)	Approximate Physical Time
5 timesteps	≈ 0.1s
10 timesteps	≈ 0.2s
20 timesteps	≈ 0.4s
50 timesteps	≈ 1.0s
100 timesteps	≈ 2.0s

To efficiently compare these configurations, a ”Quick Model” strategy was employed, in which each candidate was trained for a limited duration of 5 epochs. The performance was assessed through a dual-metric approach, evaluating both the immediate *short-term error* ($t+1$ prediction) and the *long-term error* (cumulative drift over a 50-step recursive rollout). Based on repeated measurements and a comparative analysis of the stability-to-accuracy ratio, the windows of **20 and 50 timesteps** emerged as the most effective. The 20-step window demonstrated sufficient agility for short-term dynamics, while the 50-step window provided the necessary context to mitigate long-term divergence. Consequently, these two configurations were selected as the definitive temporal bases for training the final LSTM and Transformer models.

6 Hyperparameter Tuning

Following the selection of the optimal time windows (20 and 50 timesteps), an extensive hyperparameter tuning phase was conducted to identify the best architectural configurations for both LSTM and Transformer models. This process was managed using the `keras-tuner` library, employing a search strategy aimed at minimizing the Mean Squared Error (MSE) on the validation set.

6.1 LSTM Search Space

For the LSTM architecture, the tuning focused on the depth of the network and the dimensionality of the hidden states. We utilized *Bidirectional* layers to allow the model to capture dependencies from both past and future temporal directions within the window. The search space included:

- **Number of layers**: From 1 to 3.
- **LSTM units**: Between 32 and 64 (step 32).
- **Dropout rate**: From 0.0 to 0.2, to prevent overfitting.
- **Learning rate**: Optimized among $\{10^{-2}, 10^{-3}, 10^{-4}\}$.

6.2 Transformer Search Space

The Transformer tuning focused on the capacity of the self-attention mechanism and the feed-forward networks (FFN). A custom `TimeSeriesEmbedding` was used to project the 55 physical features into the model dimension. The parameters explored were:

- **Model dimension** (d_{model}): 32 to 128 (step 32).
- **Attention heads**: 2 or 4.
- **Encoder layers**: From 1 to 4.
- **FFN dimension** (d_{ff}): 64 to 256.

The results of this search ensured that both architectures were scaled appropriately to the complexity of the quantum dynamics, balancing the number of trainable parameters with the available hardware resources.

6.3 Tuning Results

The hyperparameter optimization yielded two distinct optimal architectures for the LSTM models, reflecting the different complexities of the input windows. The structural details, including layer types and parameter counts, are summarized below.

LSTM for Window 20 For the 20-timestep window, the tuner identified a shallower architecture as the most efficient. This model consists of a single **Bidirectional LSTM** layer with 63 units per direction (totaling 126 hidden units), followed by a Dropout layer for regularization and a Dense output layer. The total number of trainable parameters is **66,961**. The simplicity of this model suggests that for shorter temporal contexts, a single recurrent pass is sufficient to capture the essential dynamics without incurring excessive computational overhead.

Table 3: Best LSTM Model Summary for Window = 20

Layer (type)	**Output Shape**	**Param #**
Bidirectional (LSTM)	(None, 126)	59,976
Dropout	(None, 126)	0
Dense	(None, 55)	6,985
Total Trainable Params:	66,961 (261.57 KB)	

LSTM for Window 50 The model optimized for the 50-timestep window exhibits a deeper hierarchy to manage the increased temporal context. The architecture features two stacked **Bidirectional LSTM** layers:

- The first layer maintains the sequence dimension with 64 units per direction (totaling 128 units).
- The second layer aggregates the sequence into a final hidden state of 64 units (32 per direction).

This configuration results in **106,231** trainable parameters. The increased depth allows the network to extract higher-level temporal abstractions, which are necessary to maintain stability and prevent drift over the longer 50-step sequence.

Table 4: Best LSTM Model Summary for Window = 50

Layer (type)	**Output Shape**	**Param #**
Bidirectional 1 (LSTM)	(None, 50, 128)	61,440
Dropout 1	(None, 50, 128)	0
Bidirectional 2 (LSTM)	(None, 64)	41,216
Dense 1	(None, 55)	3,575
Total Trainable Params:	106,231 (414.96 KB)	

The transition from a single-layer to a multi-layer architecture as the window size increases confirms that model capacity must scale with the temporal complexity of the input to ensure accurate quantum dynamics forecasting.

Transformer for Window 20 The Transformer architecture optimized for the 20-timestep window utilizes a **TimeEmbedding** layer to project the 55 input physical features into a 64-dimensional latent space[cite: 13, 14]. The structural core of the model is the **EncoderBlock**, which accounts for the vast majority of the network's capacity with 50,048 parameters, dedicated to capturing complex temporal dependencies through self-attention mechanisms[cite: 13]. To aggregate the sequence information, a **GlobalAveragePooling1D** layer is employed, followed by a hidden Dense layer with 128 units and a final output layer[cite: 13]. This configuration results in a total of **69,047** trainable parameters, offering a specialized alternative to the LSTM model with a similar memory footprint (269.71 KB) but leveraging parallelized attention-based feature extraction[cite: 13].

Table 5: Best Transformer Model Summary for Window = 20

Layer (type)	**Output Shape**	**Param #**
InputLayer	(None, 50, 55)	0
TimeEmbedding	(None, 20, 64)	3,584
Dropout	(None, 20, 64)	0
EncoderBlock	(None, 20, 64)	50,048
GlobalAveragePooling1D	(None, 64)	0
Dense (dense_3)	(None, 128)	8,320
Dropout (dropout_3)	(None, 128)	0
Dense (dense_4)	(None, 55)	7,095
Total Trainable Params:	69,047 (269.71 KB)	

Transformer for Window 50 The Transformer model configured for the 50-timestep window maintains a structural consistency with the 20-step version, demonstrating the architecture's ability to scale across different temporal resolutions without a proportional increase in parameter complexity. The model employs a **TimeEmbedding** layer to project the 50 input states into a 64-dimensional space, while the **EncoderBlock** remains the primary computational unit, capturing long-range dependencies across the extended sequence. Despite the larger input size, the total number of trainable parameters remains constant at **69,047**. This stability is achieved through the use of a **GlobalAveragePooling1D** layer, which effectively compresses the temporal dimension (None, 50, 64) into a fixed-size vector (None, 64) before the final prediction stages. This ensures that the model provides the necessary contextual depth for 50-step forecasting while remaining computationally lightweight and efficient.

Table 6: Best Transformer Model Summary for Window = 50

Layer (type)	**Output Shape**	**Param #**
InputLayer	(None, 50, 55)	0
TimeEmbedding	(None, 50, 64)	3,584
Dropout	(None, 50, 64)	0
EncoderBlock	(None, 50, 64)	50,048
GlobalAveragePooling1D	(None, 64)	0
Dense (dense_3)	(None, 128)	8,320
Dropout (dropout_3)	(None, 128)	0
Dense (dense_4)	(None, 55)	7,095
Total Trainable Params:	69,047 (269.71 KB)	

7 Model Training

The final models were trained for 15 epochs using a custom training loop to ensure precise control over the optimization process. This approach, detailed below, replaces the standard training API with a multi-stage curriculum strategy.

7.1 Custom Training Pipeline

We implemented a **Curriculum Learning** strategy divided into three progressive phases:

1. **One-Step Prediction**: Initial training on $t+1$ forecasting to stabilize the learning of basic dynamics.

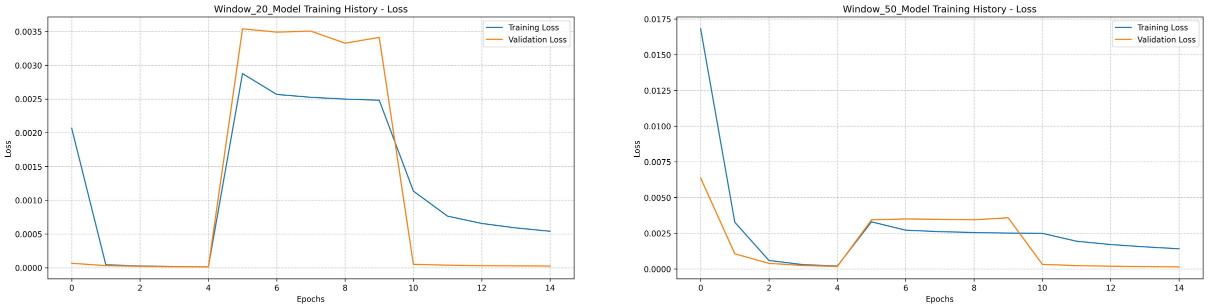
2. **Corrupted Input**: Introduction of 20% random masking to improve model robustness and prevent overfitting to noise.

3. **Autoregressive Rollout**: Recursive training over a 10-step horizon ($H = 10$) to minimize cumulative drift and ensure long-term physical consistency.

The pipeline utilized the Adam optimizer ($LR = 10^{-3}$) and Mean Squared Error (MSE) as the loss function, optimizing the models for both temporal accuracy and stability. The entire pipeline utilizes the Adam optimizer with a learning rate of 10^{-3}, specifically tuned for the 20 and 50 timestep windows. This custom loop ensures that the final weights are optimized not just for numerical accuracy, but for the physical stability of the predicted quantum evolution.

7.2 Training results

7.2.1 LSTM models

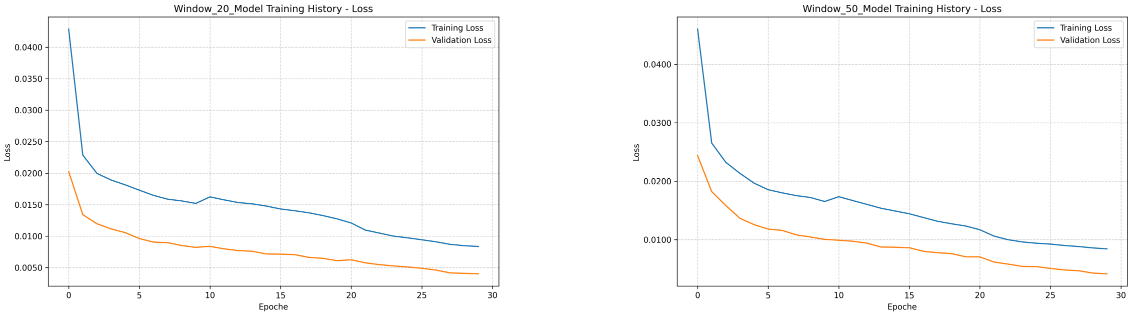


(a) Training history for Window = 20.

(b) Training history for Window = 50.

Figure 1: Comparison of training and validation loss across different input windows, highlighting the three curriculum learning phases.

7.2.2 Transformer models



(a) Training history for Window = 20.

(b) Training history for Window = 50.

Figure 2: Comparison of training and validation loss across different input windows, highlighting the three curriculum learning phases.

7.2.3 Analysis of Prediction Attention Maps

The attention maps reveal a sharp diagonal pattern, indicating that the model prioritizes recent timesteps. This behavior aligns with the physical principles of dynamical systems, where the state at time t depends primarily on immediate preceding states[cite: 5, 32]. Furthermore, we observe head specialization: one head focuses on immediate dynamics (last 3 timesteps), while another captures broader trends (5-10 timesteps). This autonomous learning of the locality principle demonstrates the Transformer's ability to discover underlying physical laws from raw data, effectively "opening" the quantum black box through interpretable mechanisms.

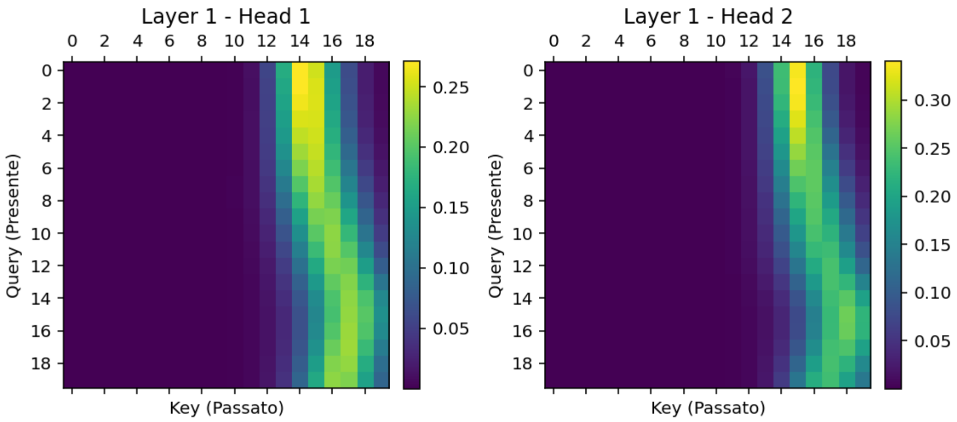


Figure 3: Attention maps - Prediction model with Window=20. Strong diagonal indicates local temporal attention.

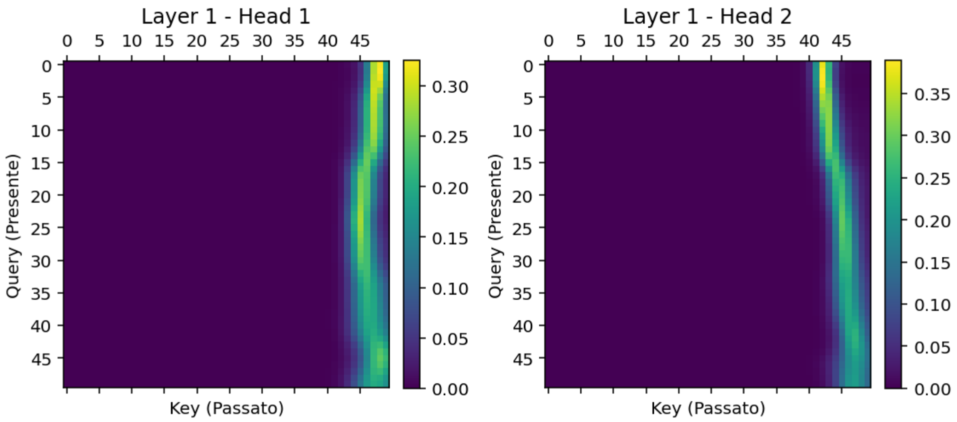


Figure 4: Attention maps - Prediction model with Window=50. Similar pattern persists at larger scale.

8 Super-Resolution

The Super-Resolution (SR) task aims to reconstruct high-resolution sequences from low-frequency sampled inputs, preserving the high-frequency oscillations of the quantum system. The architecture maps a low-resolution space (50 timesteps) to a high-resolution output (100 timesteps).

8.1 Architecture Design

The model, structured for learned upsampling and feature refinement, consists of three main stages:

- **Feature Extraction**: A `TimeSeriesEmbedding` followed by two `EncoderBlocks` captures complex temporal correlations within the input sequence.
- **Learned Upsampling**: A `Conv1DTranspose` layer learns the optimal weights for temporal interpolation, outperforming static upsampling methods.

- **Residual Refinement**: A residual connection (`Add`) combines the refined convolutional output with a linearly upsampled version of the input (`UpSampling1D`). This ensures training stability and focuses the network on learning high-frequency residuals.

8.2 Model Summary

The model comprises approximately 1.00M total parameters (334,967 trainable), balancing high capacity with computational efficiency for micro-temporal reconstruction.

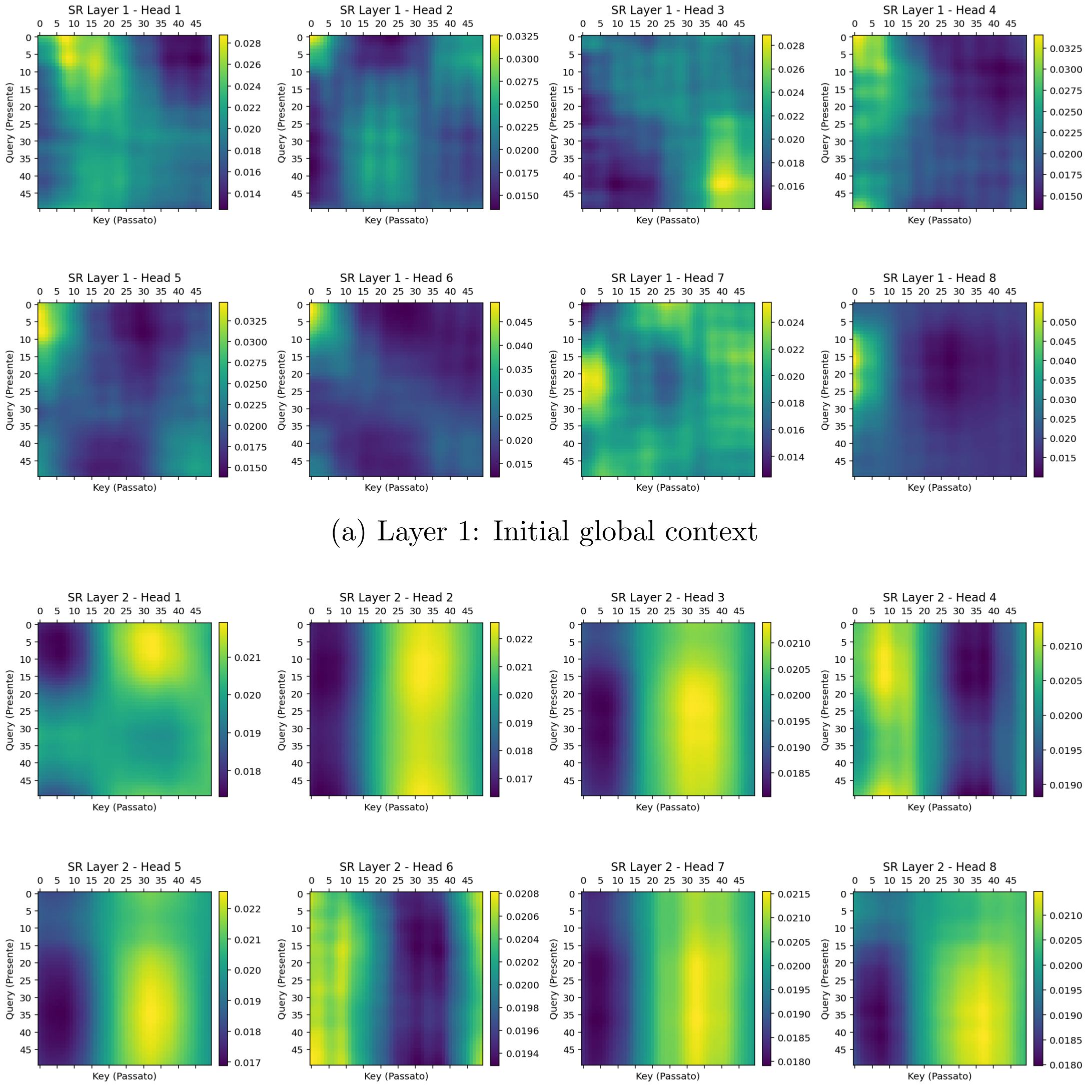
Table 7: Super-Resolution Model Layers (Input: 50 steps → Output: 100 steps)

Layer (type)	**Output Shape**	**Param #**
InputLayer	(None, 50, 55)	0
TimeEmbedding	(None, 50, 64)	3.584
Dropout	(None, 50, 64)	0
EncoderBlock1	(None, 50, 64)	149,504
EncoderBlock2	(None, 50, 64)	149,504
Conv1DTranspose	(None, 100, 64)	16,448
Conv1D	(None, 100, 64)	12,352
UpSampling1D	(None, 100, 55)	0
Dense	(None, 100, 55)	3,575
Add	(None, 100, 55)	0
Total Trainable Params:	334,967 (1.28 MB)	

8.2.1 Analysis of Super-Resolution Attention Maps

The attention maps for the Super-Resolution (SR) model exhibit fundamentally different behavior compared to the prediction architectures. Instead of a localized diagonal pattern, the SR model displays a **distributed attention** mechanism, with weights spread more uniformly across the entire sequence. This shift is physically justified by the **non-causal** nature of the reconstruction task: unlike forecasting, SR requires global context to look both forward and backward in time to accurately interpolate high-frequency details and maintain structural consistency.

To handle this increased complexity, the model employs an **8-head strategy**, allowing it to capture diverse reconstruction patterns across different frequency components and local structures. This adaptability confirms that the Transformer reconfigures its internal logic based on the objective—prioritizing causality for prediction and global context for temporal enhancement.



(a) Layer 1: Initial global context

(b) Layer 2: Refined patterns

Figure 5: Super-resolution attention maps show distributed, non-local patterns

8.3 Super-Resolution Visual Results

The visual analysis of the reconstruction confirms the high fidelity of the Super-Resolution model in restoring quantum state granularity. The SR prediction (blue line) demonstrates a near-perfect overlap with the ground truth (black dashed line), successfully interpolating the sparse information provided by the low-resolution input (red points).

Crucially, the architecture preserves both the broad global trends and the critical high-frequency oscillations across diverse signal morphologies. This robustness is evident across varied feature types, ranging from monotonic evolutions (Feature 0) and pure sinusoidal patterns (Feature 10) to more intricate, non-linear dynamics (Features 20 and 30). The model's ability to recover these micro-temporal details from sampled data validates the effectiveness of the residual refinement strategy in maintaining physical consistency.

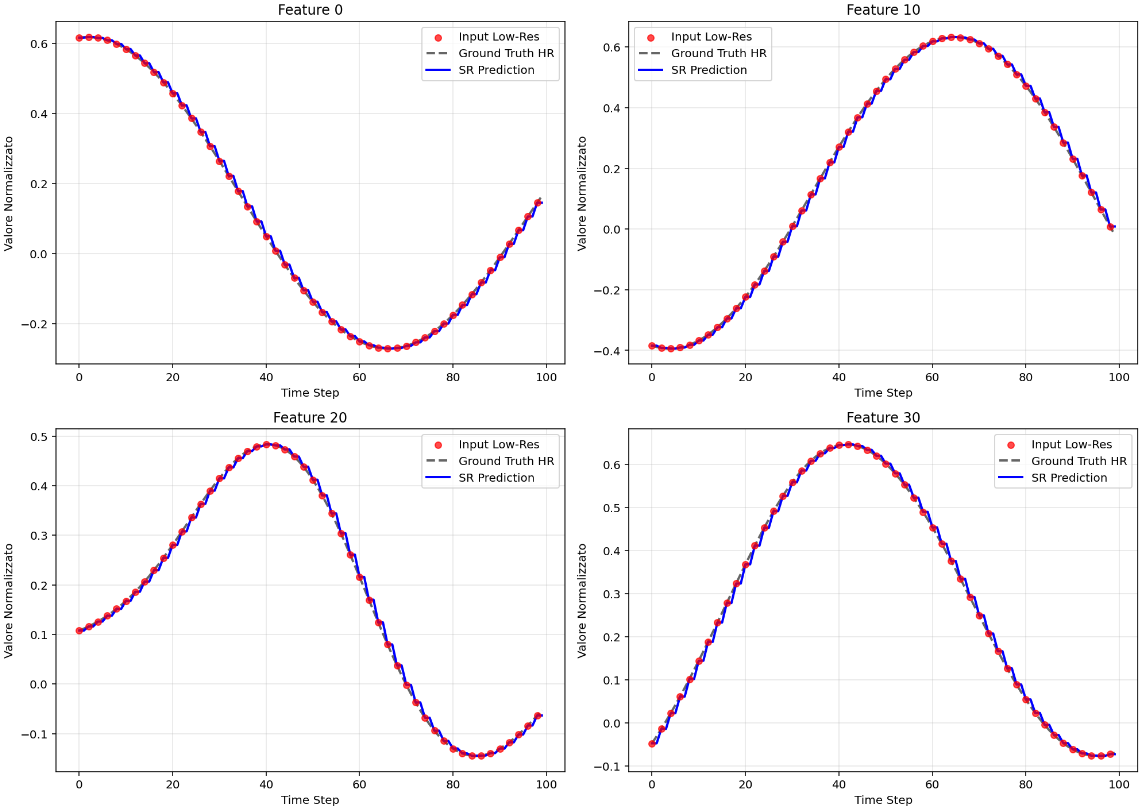


Figure 6: Super-resolution results on 4 different features. SR prediction (blue) accurately reconstructs ground truth (black) from low-res input (red dots).

9 Conclusion

This study demonstrates the efficacy of Deep Learning in modeling multi-qubit quantum dynamics, effectively bypassing the computational bottlenecks of classical simulation. By comparing **LSTM** and **Transformer** architectures, we observed that while LSTMs are proficient in capturing sequential dependencies, Transformers offer superior scalability and interpretability through **Attention Maps**, which successfully identified physical principles like temporal locality. The inclusion of a **Super-Resolution** module proved essential for restoring high-frequency oscillations, ensuring high-fidelity reconstructions from sparse data.

Our results suggest that while current models accurately track quantum trajectories, further improvements in long-term stability could be achieved by scaling model depth and expanding the temporal input windows. This hybrid framework establishes a viable path toward ”opening” the quantum black box, providing a powerful tool for the analysis and development of next-generation quantum processors.

A Hyperparameter Tuning results

A.1 LSTM Hyperparameter tested

Table 8: Table A: Hyperparameter Values for LSTM (Window: 20)

trial_id	val_loss	n_lstm_layers	lstm_units_0	dropout_0	learning_rate	lstm_units_1	dropout_1	lstm_units_2	dropout_2	tuner/epochs	tuner/initial_epoch	tuner/bracket	tuner/round	tuner/trial_id
0	2.19E-05	1	63	0	0.001	-	-	-	-	2	0	2	0	-
1	0.002453252	1	63	0.3	0.01	-	-	-	-	2	0	2	0	-
2	0.004893417	1	63	0.4	0.01	-	-	-	-	2	0	2	0	-
3	0.000889717	3	63	0.1	0.01	32	0	32	0	2	0	2	0	-
4	0.002711993	1	32	0.1	0.0001	64	0.2	64	0.1	2	0	2	0	-
5	0.002476791	1	32	0.2	0.01	32	0	32	0.1	2	0	2	0	-
6	0.001055315	2	32	0.3	0.001	64	0	64	0.1	2	0	2	0	-
7	0.006338683	2	32	0.3	0.0001	32	0.2	64	0.1	2	0	2	0	-
8	0.002486791	2	63	0.4	0.001	32	0.2	64	0.2	2	0	2	0	-
9	0.001082177	2	32	0.3	0.001	64	0	64	0.2	2	0	2	0	-
10	0.000952506	2	32	0.1	0.001	64	0.1	32	0.1	2	0	2	0	-
11	0.001817542	1	63	0.2	0.0001	32	0.1	32	0.1	2	0	2	0	-
12	1.17E-05	1	63	0	0.001	64	0	32	0.1	4	2	2	1	0
13	0.000616253	3	63	0.1	0.01	32	0	32	0	4	2	2	1	3
14	0.000719165	2	32	0.1	0.001	64	0.1	32	0.1	4	2	2	1	10
15	0.000872647	2	32	0.3	0.001	64	0	64	0.1	4	2	2	1	6
16	6.48E-06	1	63	0	0.001	64	0	32	0.1	10	4	2	2	12
17	0.000619382	3	63	0.1	0.01	32	0	32	0	10	4	2	2	13
18	1.15E-05	1	63	0	0.001	64	0.1	32	0.2	4	0	1	0	-
19	0.001959623	3	32	0	0.0001	32	0.1	32	0.1	4	0	1	0	-
20	0.002011781	2	32	0.2	0.0001	32	0.1	64	0.2	4	0	1	0	-
21	0.001147214	3	63	0.2	0.0001	64	0	32	0.1	4	0	1	0	-
22	0.001381207	3	32	0.4	0.001	64	0	32	0	4	0	1	0	-
23	0.00073007	3	63	0.3	0.001	64	0.2	64	0.1	4	0	1	0	-
24	**5.61E-06**	**1**	**63**	**0**	**0.001**	**64**	**0.1**	**32**	**0.2**	**10**	**4**	**1**	**1**	**18**
25	0.000500367	3	63	0.3	0.001	64	0.2	64	0.1	10	4	1	1	23
26	0.00054609	1	32	0.1	0.001	32	0.1	32	0.1	10	0	0	0	-
27	2.61E-05	3	63	0	0.001	32	0	32	0	10	0	0	0	-
28	0.001981181	2	63	0.4	0.001	32	0.2	64	0	10	0	0	0	-
29	0.000688632	1	63	0.2	0.001	32	0	32	0	10	0	0	0	-

Table 9: Table A: Hyperparameter Values for LSTM (Window: 50)

trial_id	val_loss	n_lstm_layers	lstm_units_0	dropout_0	learning_rate	tuner/epochs	tuner/initial_epoch	tuner/bracket	tuner/round	lstm_units_1	dropout_1	tuner/trial_id	lstm_units_2	dropout_2
0	0.000777217	2	64	0	0.0001	2	0	2	0	32	0			
1	0.000404086	3	64	0.1	0.001	2	0	2	0	64	0.1		32	0
2	0.000619548	2	32	0	0.01	2	0	2	0	64	0		64	0.1
3	0.003283719	3	32	0	0.01	2	0	2	0	32	0		64	0.2
4	0.001630283	2	32	0.2	0.0001	2	0	2	0	64	0.1		32	0.1
5	0.001130854	3	32	0.1	0.0001	2	0	2	0	64	0.2		64	0
6	0.026031587	2	64	0.2	0.01	2	0	2	0	64	0.1		64	0
7	0.002665185	3	32	0	0.001	2	0	2	0	32	0.1		32	0.2
8	0.001809371	3	32	0.1	0.0001	2	0	2	0	64	0.2		64	0.1
9	0.002633131	2	64	0	0.01	2	0	2	0	32	0.2		32	0.1
10	0.005849673	3	32	0	0.0001	2	0	2	0	64	0		32	0.2
11	0.002286975	2	32	0.2	0.001	2	0	2	0	64	0.2		64	0.1
12	0.000224314	3	64	0.1	0.001	4	2	2	1	64	0.1	1	32	0
13	0.023265254	2	32	0	0.01	4	2	2	1	64	0	2	64	0.1
14	0.000194489	2	64	0	0.0001	4	2	2	1	32	0	0	32	0.1
15	0.00071854	3	32	0.1	0.0001	4	2	2	1	64	0.2	5	64	0
16	**4.85E-05**	**2**	**64**	**0**	**0.0001**	**10**	**4**	**2**	**2**	**32**	**0**	**14**	**32**	**0.1**
17	0.000126052	3	64	0.1	0.001	10	4	2	2	64	0.1	12	32	0
18	0.033322871	1	64	0.1	0.01	4	0	1	0	64	0		32	0
19	0.000989962	3	64	0	0.001	4	0	1	0	64	0.1		64	0.2
20	0.001130223	3	64	0.2	0.01	4	0	1	0	64	0.1		32	0
21	0.002626325	2	64	0.1	0.0001	4	0	1	0	32	0.1		64	0.1
22	0.007333638	1	32	0.2	0.0001	4	0	1	0	32	0		64	0
23	0.001729255	2	64	0	0.01	4	0	1	0	64	0.1		32	0.1
24	0.000755797	3	64	0	0.001	10	4	1	1	64	0.1	19	64	0.2
25	0.001302731	3	64	0.2	0.01	10	4	1	1	64	0.1	20	32	0
26	0.000190132	1	32	0	0.0001	10	0	0	0	64	0.2		32	0.2
27	0.004638788	1	32	0.1	0.0001	10	0	0	0	64	0.1		32	0.1
28	0.001885748	1	32	0.2	0.001	10	0	0	0	64	0		32	0
29	0.000222228	3	64	0.1	0.0001	10	0	0	0	32	0		32	0

A.2 Transformer Hyperparameter tested

Table 10: Table A: Hyperparameter Values for Transformer (Window: 20)

trial_id	val_loss	dim	heads	hidden	layers	drop	lr	tuner/epochs	tuner/initial_epoch	tuner/bracket	tuner/round	tuner/trial_id
0	0.030697668	32	4	64	2	0.1	0.001	2	0	2	0	
1	0.027523527	32	4	128	1	0.1	0.01	2	0	2	0	
2	0.020675143	32	4	128	2	0.1	0.001	2	0	2	0	
3	0.062297959	64	2	64	1	0.2	0.01	2	0	2	0	
4	0.03728845	64	2	128	1	0.1	0.01	2	0	2	0	
5	0.044632785	32	4	64	1	0.2	0.01	2	0	2	0	
6	0.016369399	64	2	128	1	0.1	0.001	2	0	2	0	
7	0.024835801	64	4	64	1	0.1	0.001	2	0	2	0	
8	0.035196543	64	2	64	2	0.1	0.001	2	0	2	0	
9	0.022738226	64	2	128	1	0.2	0.001	2	0	2	0	
10	0.036709178	32	4	128	2	0.1	0.01	2	0	2	0	
11	0.062282313	64	2	64	2	0.1	0.01	2	0	2	0	
12	0.011522857	64	2	128	1	0.1	0.001	4	2	2	1	6
13	0.017350454	32	4	128	2	0.1	0.001	4	2	2	1	2
14	0.017302878	64	2	128	1	0.2	0.001	4	2	2	1	9
15	0.020626379	64	4	64	1	0.1	0.001	4	2	2	1	7
16	**0.007757253**	**64**	**2**	**128**	**1**	**0.1**	**0.001**	**10**	**4**	**2**	**2**	**12**
17	0.012083013	64	2	128	1	0.2	0.001	10	4	2	2	14
18	0.042196449	32	4	64	1	0.1	0.01	4	0	1	0	
19	0.053270955	64	4	64	1	0.2	0.01	4	0	1	0	
20	0.024851386	64	2	64	1	0.1	0.001	4	0	1	0	
21	0.018118503	64	4	128	2	0.2	0.001	4	0	1	0	
22	0.062300958	32	2	128	2	0.2	0.01	4	0	1	0	
23	0.035636406	64	2	64	2	0.2	0.001	4	0	1	0	
24	0.013776168	64	4	128	2	0.2	0.001	10	4	1	1	21
25	0.022083247	64	2	64	1	0.1	0.001	10	4	1	1	20
26	0.062296882	64	4	64	2	0.1	0.01	10	0	0	0	
27	0.010656077	64	2	128	2	0.1	0.001	10	0	0	0	
28	0.019121731	32	2	64	2	0.1	0.001	10	0	0	0	
29	0.021206487	32	2	64	1	0.2	0.001	10	0	0	0	

Table 11: Table A: Hyperparameter Values for Transformer (Window: 50)

trial_id	val_loss	dim	heads	hidden	layers	drop	lr	tuner/epochs	tuner/initial_epoch	tuner/bracket	tuner/round	tuner/trial_id
0	0.047841445	64	4	128	1	0.1	0.01	2	0	2	0	
1	0.03542519	64	2	64	2	0.1	0.001	2	0	2	0	
2	0.045364697	32	2	64	2	0.1	0.01	2	0	2	0	
3	0.062157054	64	4	128	2	0.2	0.01	2	0	2	0	
4	0.054838795	64	2	128	1	0.2	0.01	2	0	2	0	
5	0.029524777	32	4	64	1	0.1	0.001	2	0	2	0	
6	0.021758782	32	4	128	1	0.1	0.001	2	0	2	0	
7	0.04523972	64	2	128	1	0.1	0.01	2	0	2	0	
8	0.062166765	64	2	64	2	0.2	0.01	2	0	2	0	
9	0.034625404	32	2	128	1	0.1	0.01	2	0	2	0	
10	0.033417359	64	2	64	1	0.1	0.001	2	0	2	0	
11	0.023703771	64	2	128	1	0.2	0.001	2	0	2	0	
12	0.017452572	32	4	128	1	0.1	0.001	4	2	2	1	6
13	0.01826901	64	2	128	1	0.2	0.001	4	2	2	1	11
14	0.021546466	32	4	64	1	0.1	0.001	4	2	2	1	5
15	0.02837755	64	2	64	1	0.1	0.001	4	2	2	1	10
16	0.014582697	32	4	128	1	0.1	0.001	10	4	2	2	12
17	0.012628392	64	2	128	1	0.2	0.001	10	4	2	2	13
18	0.014984564	64	2	128	1	0.1	0.001	4	0	1	0	
19	0.062154893	64	2	128	2	0.1	0.01	4	0	1	0	
20	0.036083356	64	2	64	2	0.2	0.001	4	0	1	0	
21	0.027766284	32	2	64	2	0.1	0.001	4	0	1	0	
22	0.026379267	64	4	64	1	0.1	0.001	4	0	1	0	
23	0.062164418	32	4	128	2	0.2	0.01	4	0	1	0	
24	**0.010374762**	**64**	**2**	**128**	**1**	**0.1**	**0.001**	**10**	**4**	**1**	**1**	**18**
25	0.023547428	64	4	64	1	0.1	0.001	10	4	1	1	22
26	0.062174171	32	2	64	1	0.1	0.01	10	0	0	0	
27	0.062176213	64	2	64	1	0.2	0.01	10	0	0	0	
28	0.024428675	32	4	128	1	0.1	0.01	10	0	0	0	
29	0.036635246	64	4	64	2	0.2	0.001	10	0	0	0	