Higher Order Quantum Reservoir Computing

Vinamr Jain[1]

[1]**Report for Professor Romit Maulik, Pennsylvania State University**

Keywords: Quantum Machine Learning, Quantum Reservoir Computing, Lorenz-63 attractor, Chaos emulations

1 ABSTRACT

This study replicates and validates the findings presented by Tran and Nakajima (2020)[1] in their paper on Higher-Order Quantum Reservoir Computing (HQR). The objective is to showcase the efficacy of their HQR framework, which employs interconnected small quantum systems communicating through classical connections, such as linear feedback. The investigation involves the emulation of chaotic systems, including the Lorentz-63 system. The study encompasses a comprehensive comparative analysis, evaluating the Quantum Echo State Property (QESP index) and Memory Capacity (MC) across various hyperparameters. Tran and Nakajima introduce a unique quantum-innate training scheme utilizing the FORCE learning algorithm. This report replicates their results for the Lorentz-63 chaotic system and the analysis to compare HQR dynamics with conventional machine learning models such as Echo State Networks (ESN), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU).

2 THEORY AND METHOD

2.1 Introduction

Reservoir Computing (RC) conventionally employs a randomly connected network, serving as a reservoir, along with a trainable readout for pattern analysis from the reservoir's output states. The input stream undergoes nonlinear processing within the reservoir, projecting low-dimensional inputs into a high-dimensional dynamical system.

Quantum Reservoir Computing (QRC) adapts the RC paradigm by implementing the reservoir as a quantum many-body system, utilizing elements like interacting qubits or fermions driven by Hamiltonian dynamics. Quantum tunneling replaces the random connections found in classical reservoirs. The input stream drives the transition state through a unitary operator, followed by quantum measurements to obtain signals for training.

Higher-order quantum Reservoir Computing introduces a novel dimension by organizing quantum systems into an ensemble reservoir. Each system, akin to a node, receives a linear combination of common input streams and signals from other systems. This higher-order scheme enhances computational power by enabling the use of a large number of computational nodes with controllable linear feedback, thereby amplifying the system's expressiveness. Additionally, this framework facilitates the incorporation of FORCE learning and innate training in the quantum context, expanding the potential applications of QRC.

2.2 Quantum Reservoir Dynamics

Consider a one-dimensional input and output case with the input sequence $u = \{u_1, \ldots, u_L\}$ and the corresponding target sequence $\hat{y} = \{\hat{y}_1, \ldots, \hat{y}_L\}$, where u_k is a continuous variable in $[0,1]$. Quantum Reservoir Computing (QRC) emulates a nonlinear function Q to produce the output $y_k = Q(w, \rho^{(0)}, \{u_l\}_{l=1}^{k})$ $(k = 1, \ldots, L)$. Here, $\rho^{(0)}$ is the initial state of the quantum system, and w is the parameter that needs to be optimized.

A temporal learning task consists of three phases: a washout phase, a training phase, and an evaluation phase. In the washout phase, the system evolves for the first T transient steps to wash out the initial conditions from the dynamics. The training phase to optimize w is performed with training data $\{u_k\}_{k=T}^{L_1}, \{y_k\}_{k=T}^{L_1}$, where $1 \leq T < L_1 < L$, such that the mean-square error between y_k and $\hat{y}_k$ over $k = T, \ldots, L_1$ becomes minimum. The trained parameter w is used to generate outputs in the evaluation phase.

For an N-qubits system, at time $t = (k-1)\tau$, the input $u_k \in [0,1]$ is fed to the system by setting the density matrix of the first spin to

$$\rho_{u_k} = (1-u_k)|0\rangle\langle 0| + u_k|1\rangle\langle 1| \in \mathbb{M}_{2\times 2}.$$

Therefore, the density matrix $\rho \in \mathbb{M}_{2^N \times 2^N}$ of the entire system is mapped by a completely positive and trace preserving (CPTP) map

$$\rho \to \mathscr{T}_{u_k}(\rho) = \rho_{u_k} \otimes \mathrm{Tr}_1[\rho],$$

where Tr_1 denotes a partial trace with respect to the first qubit. After the input is set, the system continues evolving itself during the time interval τ. The dynamics are governed by the Schrödinger equation, and the information of the input sequence encoded

in the first spin spreads through the system. It follows that the state of the system before the next input u_{k+1} is

$$\rho^{(k)} = e^{-iH\tau}\mathscr{T}_{u_k}(\rho^{(k-1)})e^{iH\tau},$$

where $\rho^{(k)} = \rho(k\tau)$ is the density matrix at $t = k\tau$.

If we employ the ordered basis $\{O_j\}$ in the operator space, then the observed signals at time t are the first N_{out} elements

$$s_j(t) = \text{Tr}[\rho(t)O_j],$$

where the selection of observables depends on the physical implementation of the system.

2.3 Temporal Multiplexing

A temporal multiplexing scheme is introduced to enhance performance in extracting dynamics. In this scheme, signals are measured not only at time $k\tau$ but also at each of the subdivided V time intervals during the evolution in the interval τ to construct V virtual nodes. The density matrix is then updated by

$$\rho((k-1)\tau + \frac{1}{V}\tau) = U_{(\frac{\tau}{V})}T_{u_k}(\rho^{(k-1)})U^{\dagger}_{(\frac{\tau}{V})},$$

$$\rho\left((k-1)\tau + \frac{v}{V}\tau\right) = U_{(\frac{\tau}{V})}\rho\left((k-1)\tau + \frac{v-1}{V}\tau\right)U^{\dagger}_{(\frac{\tau}{V})}, \quad (v = 2,\ldots,V),$$

where $U_{(\tau/V)} = e^{-iH(\tau/V)}$. Therefore, temporal signals from $N_{\text{out}}V$ nodes can be obtained.

The learning procedure is straightforward, parameterizing the linear readout function as $y_k = \sum_{i=0}^{N_{\text{out}}V} w_i x_{ki}$, where $x_{ki} = s_j((k-1)\tau + \frac{v}{V}\tau)$ for $i = (j-1)V + v > 0$ ($1 \le v \le V, 1 \le j \le N_{\text{out}}$). Here, $x_{k0} = 1.0$ are introduced as constant bias terms, and $w = [w_0, w_1, \ldots, w_{N_{\text{out}}V}]$ represents the readout weight parameters. If we denote K as the number of time steps used in the training phase, w is optimized via linear regression, or Ridge regression in the matrix form

$$\hat{w}^T = (X^T X + \beta I)^{-1} X^T \hat{y},$$

where $\hat{y} = [y_1, \ldots, y_K]^T$ is the target sequence, $X = [x_{ki}] \in \mathbb{R}^{K\times(N_{\text{out}}V+1)}$ is the training data matrix, and β is the parameter serving as the positive constant shifting the diagonals introduced to avoid the problem of the near-singular moment matrix.

2.4 Higher Order Quantum Reservoir Computing model

In the paper, The fully connected transverse field Ising model, a standard framework for building Quantum Reservoirs (QR), is employed. The Hamiltonian is given by $H = J\sum_{i\neq j} h_{i,j}\sigma_i^x\sigma_j^x + J\sum_j g_j\sigma_j^z$, where σ_j^γ ($\gamma \in \{x,y,z\}$) is the operator measuring the spin j along the γ direction. This operator can be described as an N-tensor product of 2×2-matrices:

$$\sigma_j^\gamma = I \otimes \ldots \otimes \sigma_j^\gamma \otimes \ldots \otimes I,$$

where $I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $\sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, $\sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$, and $\sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$. J is the coupling magnitude of the Hamiltonian, while the coupling parameter $h_{i,j}$ and the transverse field parameter g_j are uniformly selected from the range $[-1.0, 1.0]$. $N_{\text{out}} = N$ observables $O_j = \sigma_j^z$ are chosen to produce the signals of the readout nodes. A Higher-Order Quantum Reservoir (HQR) comprises an ensemble of N_{qr} Quantum Reservoirs (QRs). The l-th system Q_l has the Hamiltonian H_l with N_l qubits (known as true nodes) and V_l virtual nodes. To feed an N_{in}-dimensional external input u into the HQR, u is transformed into an N_{qr}-dimensional input via the linear transformation $u \to W_{\text{in}}u$, where $W_{\text{in}} \in \mathbb{R}^{N_{\text{qr}}\times N_{\text{in}}}$ is fixed randomly.

The reservoir states of Q_l at time $t = k\tau$ ($k \ge 0$) are represented by a $V_l N_l$-dimensional vector z_{kl}, initialized at $k = 0$ as the zero vector z_0. Then, the reservoir states of the HQR at $t = k\tau$ can be represented by an N_{total}-dimensional vector $z_k = [z_{k1}^T, \ldots, z_{kN_{\text{qr}}}^T]^T$, where $N_{\text{total}} = \sum_{l=1}^{N_{\text{qr}}} N_l V_l$. The scaling $z \to (z+1)/2$ is employed, such that the elements of z are in $[0,1]$.

At $t = (k-1)\tau$ ($k \ge 1$), the mixed input u'_{kl}, injected into Q_l, is a linear combination of the external input and the reservoir states from all QRs. The mixed input $u'_k \in \mathbb{R}^{N_{\text{qr}}\times 1}$ can be represented in matrix form as

$$u'_k = (1-\alpha)W_{\text{in}}u_k + \alpha W_{\text{con}}z_{k-1},$$

where $W_{\text{con}} \in \mathbb{R}^{N_{\text{qr}}\times N_{\text{total}}}$ is randomly generated and fixed to represent the linear feedback connection between the QRs, and α ($0 \le \alpha \le 1$) is defined as the connection strength parameter.

After injecting the mixed input u'_{kl} into Q_l, the density matrix in Q_l is transformed by the CPTP map, $T_{u'_{kl}}$, and consequently evolves in each τ/V time.

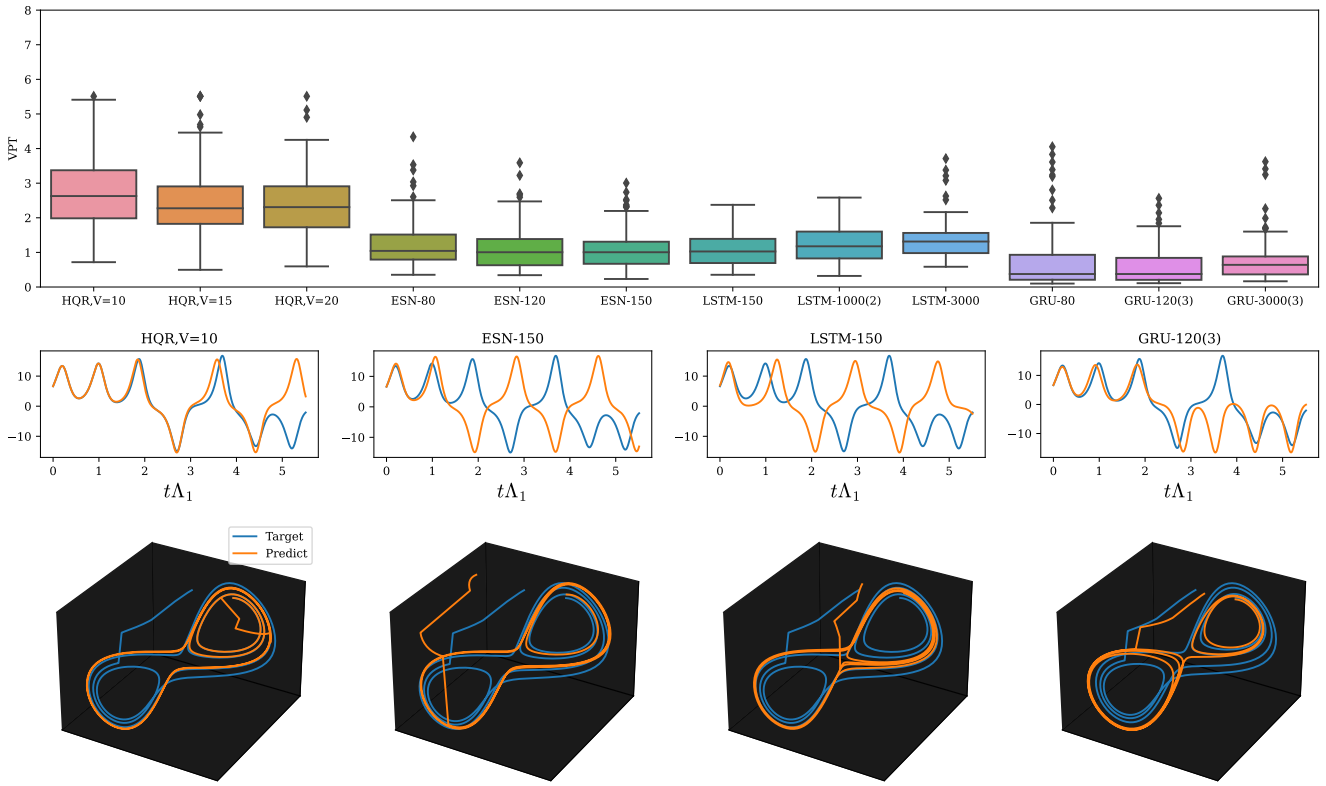

Figure 1. The box-and-whisker plot is a graphical representation of statistical data that displays the interquartile range within the box. The whiskers represent the 5th and 95th percentiles, while the line across the box represents the median. At the bottom of the plot, there are typical predicted time series of the Lorenz system based on 1000 time steps during training over 100 random predictions. The time scale of the series is normalized using the Lyapunov time Λ_1^{-1}. In our study, the maximum Lyapunov exponent of the Lorenz attractor system is $\Lambda_1 = 0.9056$

2.5 Lorenz System Emulation

To evaluate the prediction performance, we compute the valid prediction time VPT $= \Lambda_1^{-1} \arg\max_{t_f}\{NRMSE(y_t) \leq \varepsilon, \forall t \leq t_f\}$, which is the largest time t_f (normalized with respect to the maximum Lyapunov exponent Λ_1 of the chaotic system) where the Normalized Root Mean Square Error (NRMSE) is smaller than ε ($\varepsilon = 0.5$ in our experiments). A large VPT indicates a long prediction horizon in the model's performance.

The Lorenz attractor is described by three ordinary differential equations:

$$\frac{dx}{dt} = a(y-x), \quad \frac{dy}{dt} = x(b-z)-y, \quad \frac{dz}{dt} = xy - cz,$$

where $(a,b,c) = (10, 28, 8/3)$.

3 RESULTS

Figures 1, 2 & 3 demonstrate the distribution of the VPT over 100 random tests for the HQR model, and the three best results in each of the other models. Note that n in HQR-n denotes an HQR with n QRs (Here, we use 5). Here in HQR-5, each node comprises of 6 qubits, with τ = 4.0 (interval between inputs), Coupling strength J = 2.0, V $\in \{10, 15, 20\}$ corresponding to a system with $6 \times 5 \times$V nodes. The dimension of the input data is 1, the Transient time step length is 2000, and the Ridge (using pseudo inverse) parameter used is 10^{-7}. Notice that irrespective of the number of training time steps, the HQR outperforms other models even with a small number of computational nodes.

Figure 4 illustrates VPT of HQR models with varying connection strength α. Here we use 15 virtual nodes (V) in each model. We observe that the model with the α =0.0 (without the feedback scheme) outperforms other models with higher α's.

Figure 5 shows how the performance of different models varies with the number of time steps (K) for training. For the HQR-5 models, α used is 0.0 and V=15. Figure 6 illustrates the VPT of newly trained models with varying parameters. I'm not sure what the bottom figures are supposed to indicate.

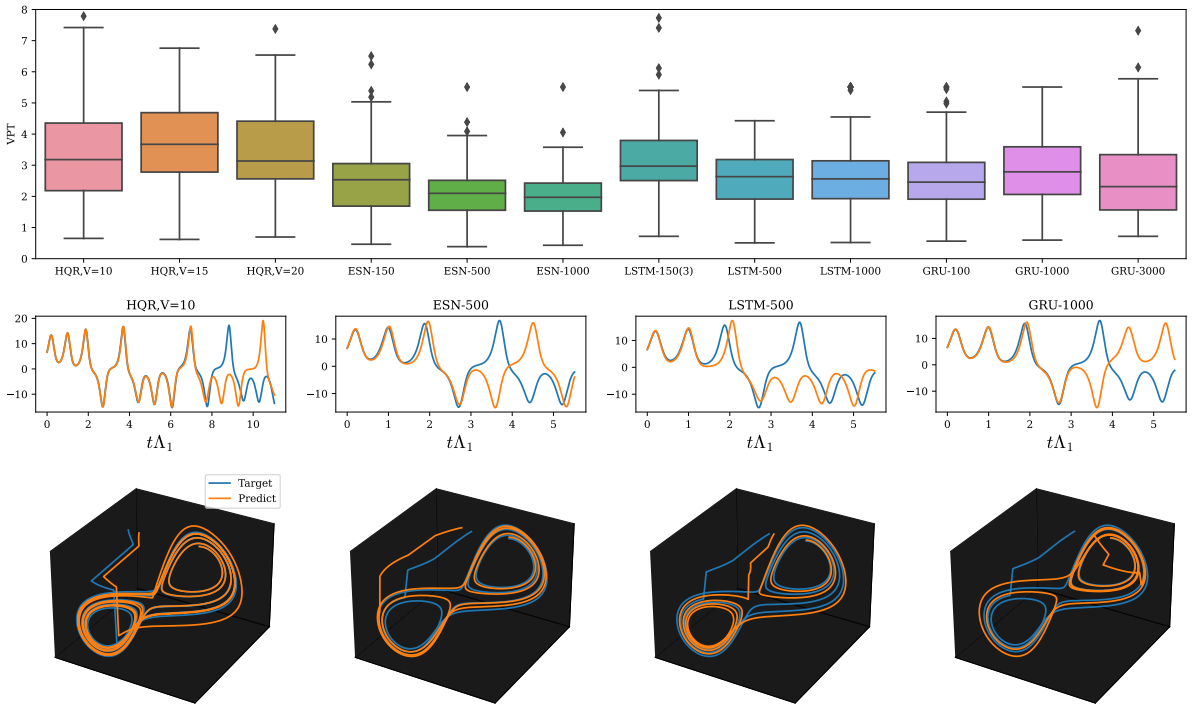

Figure 2. The box-and-whisker plot is a graphical representation of statistical data that displays the interquartile range within the box. The whiskers represent the 5th and 95th percentiles, while the line across the box represents the median. At the bottom of the plot, there are typical predicted time series of the Lorenz system based on 10^4 time steps during training, over 100 random predictions. The time scale of the series is normalized using the Lyapunov time Λ_1^{-1}. In our study, the maximum Lyapunov exponent of the Lorenz attractor system is $\Lambda_1 = 0.9056$

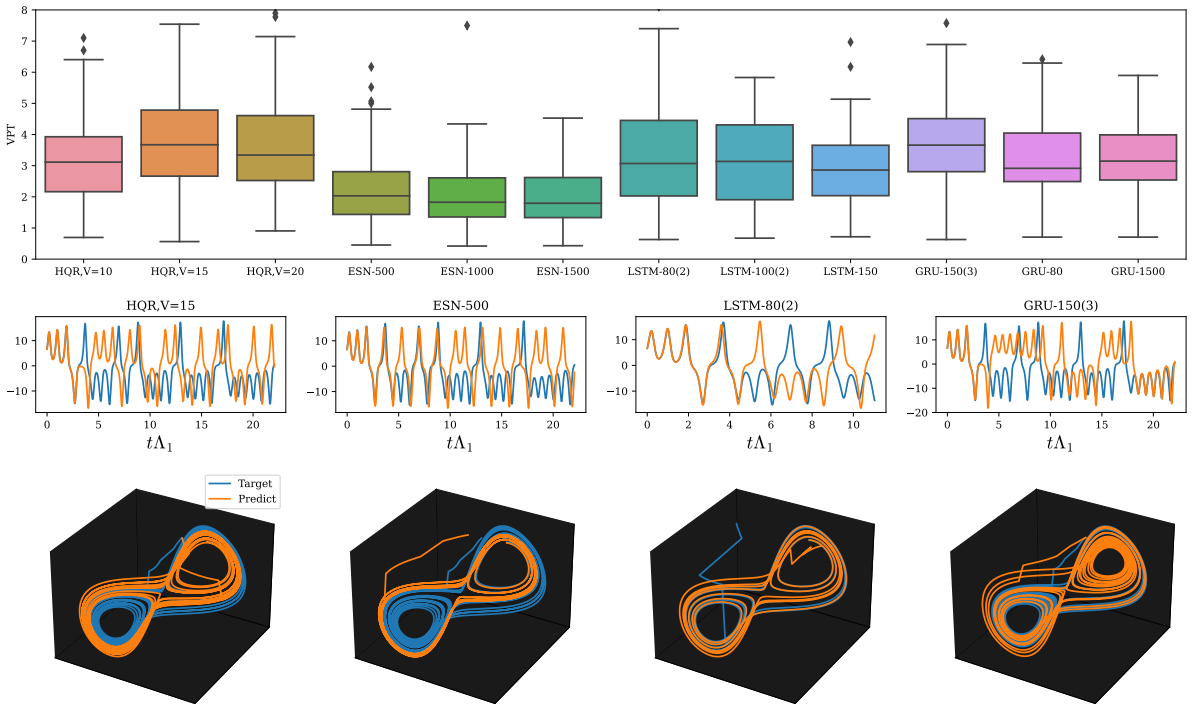

Figure 3. The box-and-whisker plot is a graphical representation of statistical data that displays the interquartile range within the box. The whiskers represent the 5th and 95th percentiles, while the line across the box represents the median. At the bottom of the plot, there are typical predicted time series of the Lorenz system based on 10^5 time steps during training over 100 random predictions. The time scale of the series is normalized using the Lyapunov time Λ_1^{-1}. In our study, the maximum Lyapunov exponent of the Lorenz attractor system is $\Lambda_1 = 0.9056$

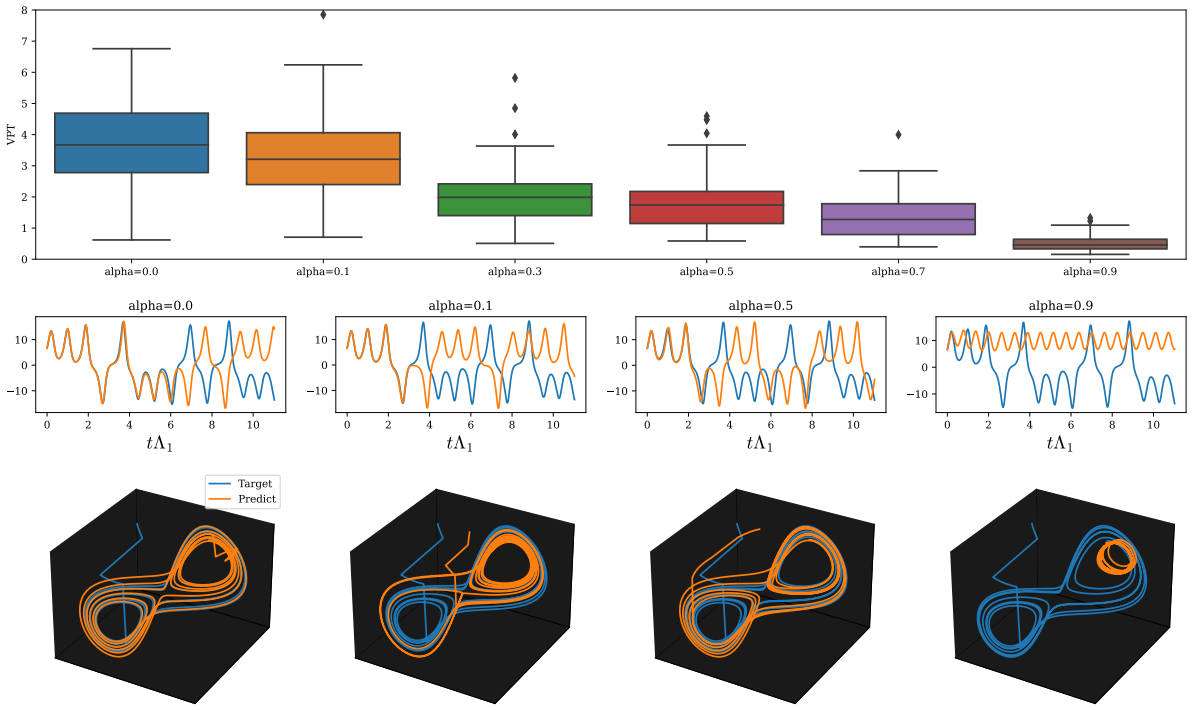

Figure 4. Box plots show the distribution of VPT for 100 predictions with 10^4 time steps, varying connection strengths α, with target and predicted trajectories.

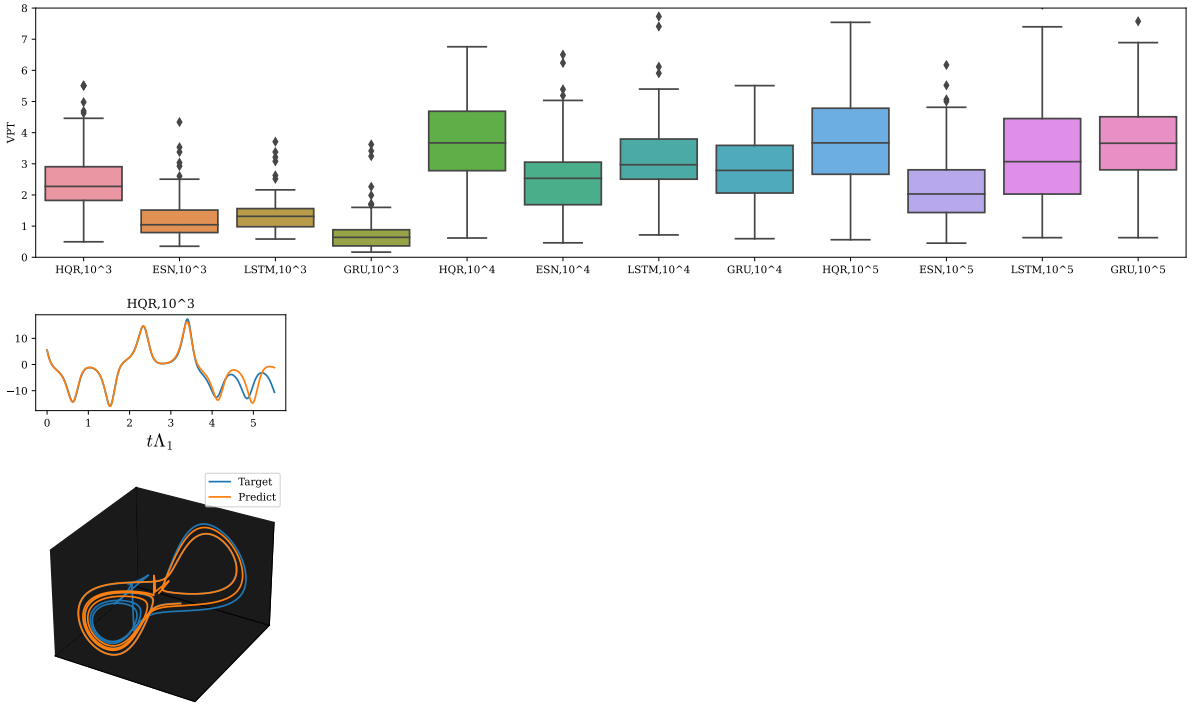

Figure 5. The performance of prediction models due to the number of time steps for training.

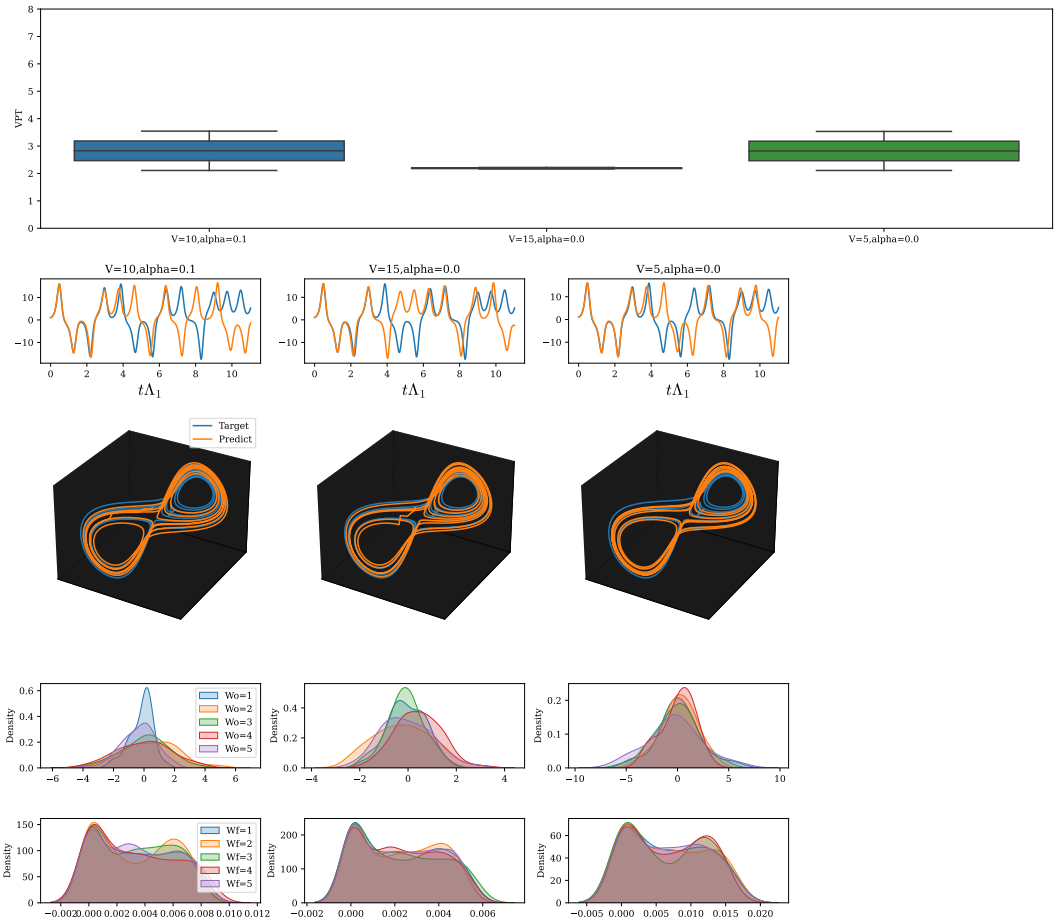

Figure 6. The performance of newly trained models with different V and α's. The models are trained over 10^4 timesteps over 2 random prediction samples.

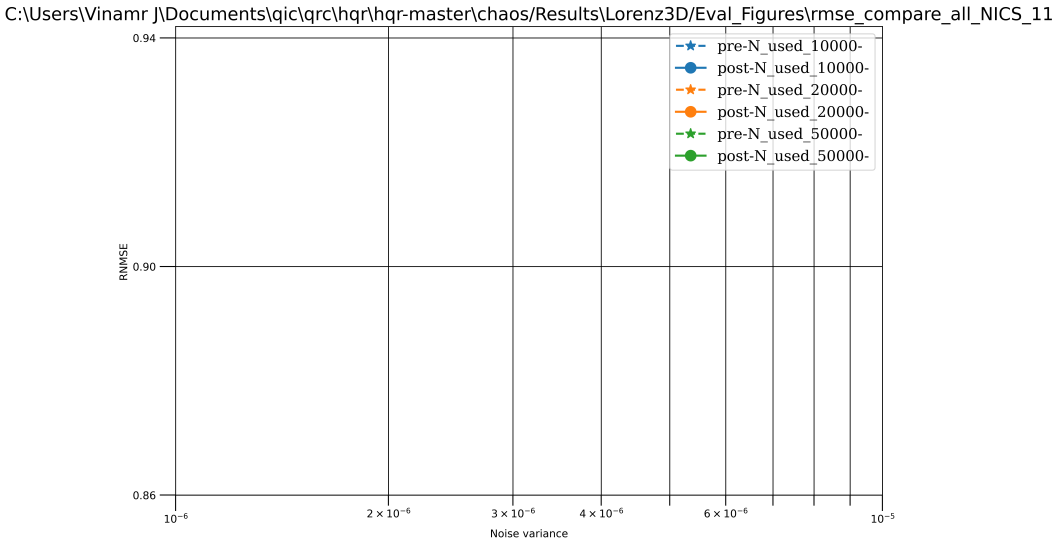

Figure 7. (What is supposed to be) The NRMSE values averaged over 11 random predictions on the Lorenz system along different levels of noise added to the reservoir signals. The NRMSE values are evaluated in two regimes: without innate training (pre-train, dashed lines) and with innate training (post-train, solid lines). The number K of time steps used in the main training are 10^4, 2×10^4, and 5×10^4

```
Total number of parameters: 82096
SAVING MODEL...
Recording time...
Total training time is 817.6245226860046
MEMORY TRACKING IN MB...
Script used 139.05078125 MB
SAVING MODEL...
RANDOM SEED: 0
Reference train time 84600.0 seconds / 1410.0 minutes / 23.5 hours.
```

Figure 8. Training time and memory utilisation for training a model with 150 computational nodes, with 10^4 training time steps

4 DISCUSSION

We tried plotting the NRMSE values of the quantum innate training model against the noise variance; however, for some reason, it shows nothing, as can be seen in Figure 7. Figures 1 through 5 were plotted using pickled pre-trained models, while 6 uses newly trained models. 8 shows the training time and memory used for the training of a typical HQR-5 model. In addition to the plots, using the script sort_model_test_vpt.py, we were able to sort the models in decreasing order of VPT (50th percentile). We find that in the case of K (number of time steps)=1000, the model with hype-parameters $\alpha = 0.1$, V=10 and ridge parameter $\beta = 10^{-11}$ has the highest VPT. For K= 10^4, model with $\alpha = 0.0$, V=15 and $\beta = 10^{-7}$ performs the best, while for K=10^5, $\alpha = 0.0$, V=15 and $\beta = 10^{-9}$ works the best. For the case of K=10^5, we notice that other models like GRUs and LSTMs don't trail behind by much, compared to models with lower Ks where HQRs dominate. The corresponding .txt files have been attached alongside this report.

REFERENCES

[1] Tran, Q. H. and Nakajima, K. (2020). Higher-order quantum reservoir computing. https://github.com/OminiaVincit/higher-order-quantum-reservoir.