Advancing Seismic Risk Prediction with Quantum-Inspired and Hybrid Classical–Quantum Deep Learning

Su Fong Chien,[1,*] Samuel Yen-Chi Chen,[2,†] Heng Siong Lim,[3,‡] and Charilaos C. Zarakovitis[1,§]

[1] *Axon Logic, M. Timotheou 21, 142 31 Athens, Greece*

[2] *Computational Science Initiative, Brookhaven National Laboratory, Upton, NY 11973, USA*

[3] *Faculty of Engineering and Technology, Multimedia University, 75450 Melaka, Malaysia*

(Dated: February 4, 2026)

Abstract

Traditional fragility curves and classical machine learning models, such as random forests, are limited in predicting maximum structural responses at the individual building level due to weak correlations between structural parameters and seismic record characteristics. Although long short-term memory (LSTM) models demonstrate strong predictive capabilities, they remain computationally demanding. To address this, the study evaluates pure LSTM and Bi-LSTM models, LSTM variants augmented with Matrix Product State (MPS) or Tensor Train structures (MPS-LSTM), and a hybrid CNN–QLSTM–LSTM architecture that combines CNN-based local feature extraction, quantum LSTM (QLSTM) gating for enhanced feature encoding, and classical temporal modeling, all aimed at improving efficiency and feature expressivity for structural response prediction. The MPS functions not only as an effective feature compressor but also as a nonlinear entangling pre-processor that reorganizes and strengthens feature correlations. Event-based sequences with LSTM memory reset ensure physically consistent learning within individual records. Seismic datasets are carefully preprocessed into non-overlapping event-based sequences, with each LSTM sequence reset to avoid temporal leakage across records. Zero-padding is applied only with proper masking, ensuring that non-physical timesteps are excluded from loss computation. The models employ a sequence-to-sequence prediction strategy rather than a sequence-to-one setup, allowing the network to capture the full temporal evolution of structural responses. Unlike prior autoregressive approaches, the proposed models do not require historical response data yet achieve accurate predictions. A comprehensive evaluation of all solutions was conducted, and model selection was guided by statistical metrics such as test RMSE, R^2, and SMAPE through comparison of predictions against the ground truth. Our findings indicate that Bi-LSTM, MPS-LSTM, and CNN–QLSTM–LSTM architectures all yield strong models. However, their selection involves trade-offs when considering the previously discussed metrics as well as energy consumption, which is closely tied to the total number of trainable parameters. The results confirm that classical models like Bi-LSTM and quantum-inspired methods such as MPS-LSTM can be reliably trained on modern GPU hardware. Additionally, the hybrid CNN–QLSTM–LSTM model demonstrates that classical–quantum integration is practical for seismic regression tasks using today's hybrid computing systems. In the long term, fully quantum implementations may further accelerate both training and prediction, enabling scalable and accurate seismic vulnerability assessment for individual buildings.

* sufong.chien@axonlogic.gr

† ycchen1989@ieee.org

‡ hslim@mmu.edu.my

§ c.zarakovitis@axonlogic.gr

I. INTRODUCTION

Long Short-Term Memory (LSTM) networks, a type of recurrent neural network (RNN) introduced by Hochreiter and Schmidhuber in 1997 [1], have demonstrated remarkable capability in modeling nonlinear temporal dependencies across diverse domains of time-series data. In the financial sector, LSTMs have been widely applied for stock price forecasting, volatility modeling, and algorithmic trading, achieving superior performance compared to traditional autoregressive and feedforward models due to their ability to capture long-range dependencies and noise patterns in market sequences [2]. In the seismic and structural engineering domain, LSTM-based architectures have been explored to approximate the nonlinear dynamics of structural systems and reduce computational complexity in model order reduction [3]. This research integrates autoencoder and LSTM networks to construct a data-driven reduced-order model capable of capturing the essential dynamics of nonlinear structural responses under seismic excitations, achieving promising accuracy yet at significant computational cost. More recently, Chien et al. extended the concept of LSTM into the quantum domain by proposing a hybrid Quantum LSTM (QLSTM) model that combines classical recurrent processing with variational quantum circuits to enhance feature entanglement and learning efficiency, demonstrating the method's potential in high-dimensional signal processing tasks such as indoor localization [4]. These applications highlight the versatility and robustness of LSTM architectures in handling temporal correlations and dynamic dependencies across various scientific and engineering disciplines.

One potential bridge between classical deep learning and quantum-inspired computation is tensor networks, such as the **Matrix Product State (MPS)**, also known as the **Tensor Train (TT)** decomposition in applied mathematics. Originally developed in quantum physics to efficiently represent high-dimensional quantum states, MPS has gained attention in machine learning as a powerful tool for reducing model complexity while preserving essential information. In particular, integrating MPS with deep learning architectures, such as **Long Short-Term Memory (LSTM)** networks, offers a novel approach to enhancing computational efficiency and scalability. Rose Yu et al. introduced Tensor-Train RNNs and Higher-Order Tensor RNNs for long-term forecasting, using tensor-train decomposition to model higher-order dynamics efficiently and achieve improved long-horizon predictions [5]. More recently, Moore et al. proposed MPSTime, a matrix-product-state-based framework

that captures complex temporal correlations for time-series tasks such as classification and imputation, offering both efficiency and interpretability [6].

In this study, we carefully process millions of rows of raw feature data. To enforce physical consistency, time-series segmentation is performed separately within each (AnalysisNo, FrameName, EQ_Record_Name) group, ensuring that no sequence spans across different buildings or earthquake input histories. We employ the BLOCK windowing technique to construct non-overlapping windows of 120 time steps, strictly bounded within individual records. The study compares LSTM, Bi-LSTM, MPS-LSTM, and a hybrid CNN–QLSTM–LSTM model that fuses convolutional, quantum, and classical recurrent layers to enhance efficiency and feature expressivity. The insights gained from this work contribute to the ongoing development of computationally efficient and scalable machine learning frameworks, with potential applications in both classical and emerging quantum machine learning paradigms.

II. DATA PRE-PROCESSING

A. Data Preparation and Physically Consistent Sequence Modeling

In this study, we place strong emphasis on developing a physically consistent data preprocessing and sequence modeling framework to ensure that the learning process adheres to the fundamental characteristics of seismic structural responses. In contrast to previous studies that incorporated one-step or two-step future response values into their model inputs, thereby introducing the risk of temporal information leakage [7], the present study refrains from using any target displacement, velocity, or acceleration data as predictive features. Instead, the model relies exclusively on measured sensor responses and derived structural parameters that are physically observable or computable at the current and past timesteps. This ensures that the predictive framework maintains causal validity and remains fully deployable for real-time inference.

B. Event-wise Data Integrity and Non-overlapping Sequence Construction

Each input sequence represents a complete structural response history under a distinct earthquake excitation. From a physics standpoint, such sequences are mutually independent,

as they correspond to different ground motion records and structural configurations. To prevent the model from inferring spurious temporal continuity across unrelated seismic events, we enforce strict grouping based on event identifiers, defined by the combination of AnalysisNo, FrameName, EQ_Record_Name. No data window crosses these event boundaries, ensuring that the LSTM memory state is reset at the start of each independent sequence, consistent with the physical constraint that structural hysteresis and damage accumulation are event-specific. This approach eliminates the possibility of inter-event information leakage, a common pitfall in time-series learning for seismic data. Furthermore, no overlapping windows are used during training. Furthermore, our non-overlapping segmentation preserves the authenticity of each response segment and maintains one-to-one correspondence between data samples and physical events.

C. Marking and Masking Strategy for Sequence Padding

Because the lengths of seismic response records vary across events, we adopt zero-padding with explicit masking to standardize input dimensions for batch processing. Zero-padding introduces artificial timesteps with no physical meaning; therefore, a sequence mask is applied throughout training and validation so that padded entries are completely excluded from the loss computation and gradient propagation. This ensures that the model learns only from physically valid timesteps. The mask mechanism is integrated in both training and validation functions, guaranteeing consistent handling of variable-length sequences across all stages.

D. Sequence-to-Sequence Prediction Framework

The proposed network adopts a sequence-to-sequence (Seq2Seq) LSTM architecture, which outputs predicted responses at every timestep of an input sequence. This differs from the conventional sequence-to-one formulation that commonly used in earlier studies—which predicts only the final or next-step value. The Seq2Seq design offers two main advantages:

- It allows the model to capture evolving temporal dependencies over the entire dynamic response, particularly important in nonlinear or damage-sensitive regimes.

- It provides fine-grained timestep-level predictions, enabling detailed comparison with recorded ground-truth displacement histories and facilitating interpretability of the temporal evolution of errors.

III. LONG-SHORT TERM MEMORY NETWORKS

A time series refers to a collection of data points arranged chronologically, depicting the evolution of a particular metric over time. There are few examples can be easily seen in our daily life such as stock prices over days/months/years, traffic volume on roads, highways, or public transportation systems varies throughout the day, week, and year, recording sensor data consistently over time, and etc. What distinguishes time series data is its capacity to utilize historical data to forecast future trends. LSTM represents a type of recurrent neural network (RNN) designed to grasp order dependencies inherent in sequence prediction tasks [8]. Its proficiency in retaining past information renders it particularly effective in forecasting stock prices. This efficacy stems from the fact that predicting future stock prices inherently relies on understanding and incorporating patterns from previous price movements. LSTM networks are specifically engineered to address the challenge of long-term dependencies encountered in traditional RNNs, which often suffer from the vanishing gradient problem. An LSTM unit maintains a cell state, which represents the network's current long-term memory and captures information from previous time steps. At each time step, the LSTM also receives the previous hidden state, which serves as a contextual input, and the current input data from the sequence. Each LSTM unit comprises four gates/components, each with a distinct function:

- **Forget Gate**: Determines which portions of the long-term memory are to be disregarded, considering inputs from the previous hidden state and the current input data.

- **Input Gate**: Utilizes a sigmoid activation function to identify relevant components of new information to retain.

- **Update Gate**: Generates candidate values to modify the cell state; these values are filtered by the input gate before being added.

- **Output Gate**: Generates a new hidden state by amalgamating the updated cell state, the previous hidden state, and the new input data.

In summary, LSTMs differ from traditional feedforward neural networks because of their feedback connections. This characteristic allows LSTMs to analyze entire sequences of data, such as time series, by preserving relevant information about preceding data points. Instead of treating each point in the sequence in isolation, LSTMs leverage past data to aid in processing new data points. Consequently, LSTMs excel at handling sequential data types like text, speech, and general time series, as they can effectively capture and utilize dependencies between successive elements in the sequence. This architecture of LSTM is depicted in Figure 1(a). Figure 1(b) illustrates the quantum LSTM (QLSTM). In this formulation, the inputs to the σ functions in equations (1a)–(1c) and (1e) are replaced by $VQC_1 - VQC_3$, and VQC_4, respectively. Meanwhile, the outputs on the right-hand side of equations (1f) and (1g) serve as the inputs to VQC_5 and VQC_6.

The functionality of LSTM node is based on Eq(1) below:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{1a}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{1b}$$

$$\widetilde{C}_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{1c}$$

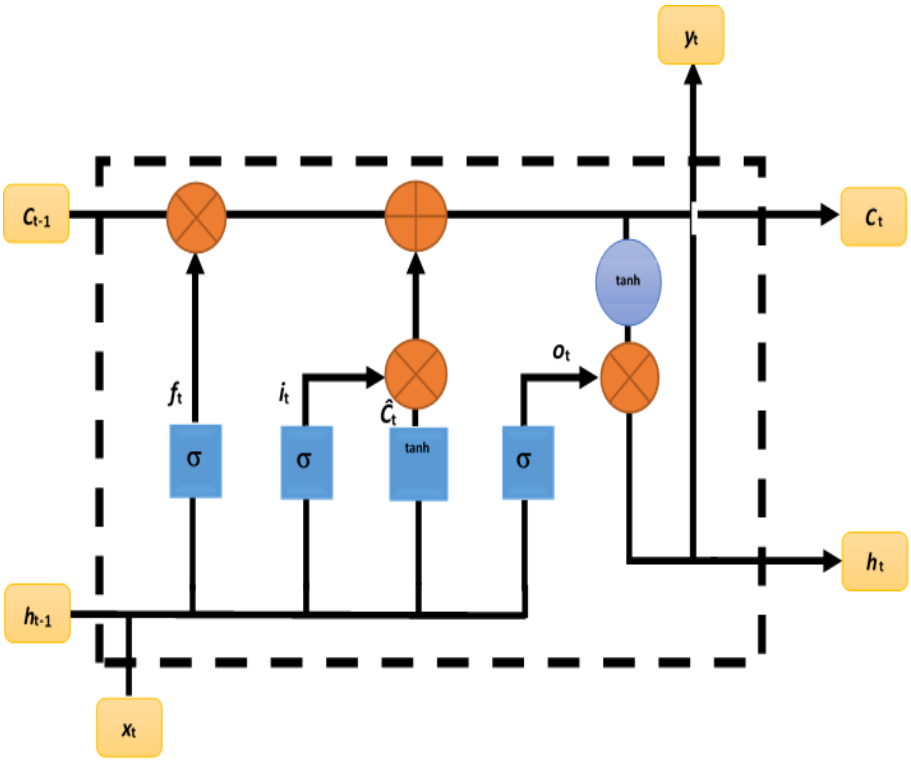
$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t \tag{1d}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{1e}$$

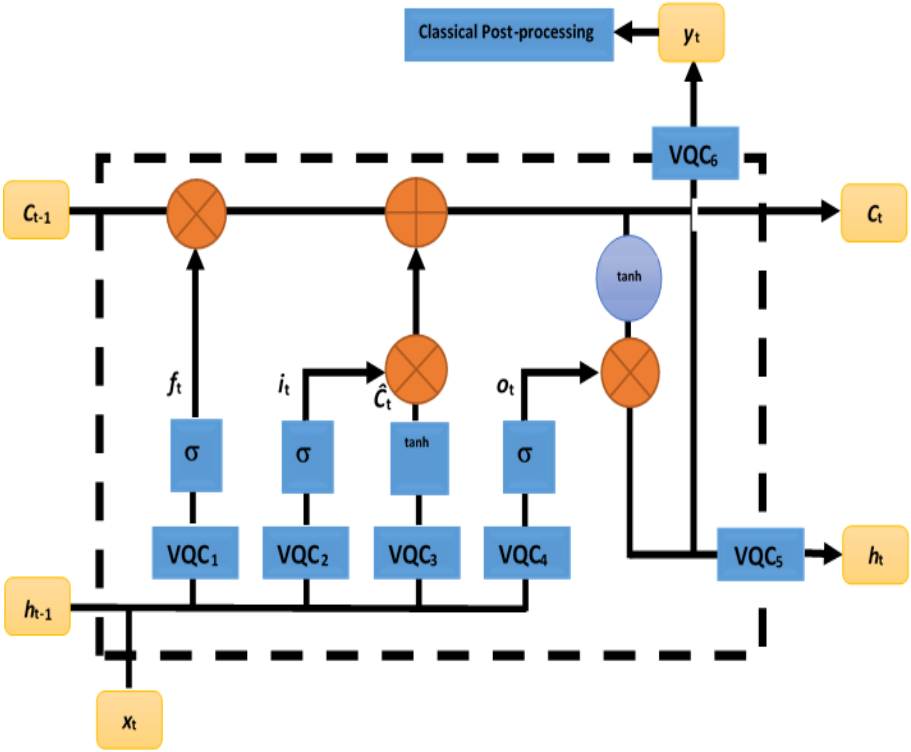
$$h_t = o_t * tanh(C_t) \tag{1f}$$

$$y_t = o_t * tanh(C_t) \tag{1g}$$

where x_t is the input data at time t, y_t is the output, h_t is the hidden state, C_t is the cell state. σ and tanh blocks denotes the sigmoid and hyperbolic tangent activation function, respectively. Note that $\otimes$ and $\oplus$ represent point-wise multiplication and addition, respectively. The sigmoid function plays a crucial role in the **forget gate layer** of an LSTM. By restricting outputs to the range of [0, 1], it signifies the extent to which information should be discarded (0) or retained (1) from the previous cell state. The forget gate examines h_{t-1} and x_t and produces a value between 0 and 1 for each element in the cell state C_t. The subsequent step involves determining the new information to be incorporated into the cell



(a) Schematic of the LSTM node.



(b) Schematic of the QLSTM node.

FIG. 1: Internal structure illustration for both LSTM and QLSTM

state. This process consists of two interconnected parts in which the sigmoid gate layer determines precisely what modifications to make within a range of 0 to 1 and the tanh gate layer generates candidate values that could potentially be added to the cell state within a range of -1 to +1. Afterward, the cell state undergoes an update using the information obtained from the preceding two processes. In this updating processes, the incoming cell state C_{t-1} is adjusted to incorporate the decisions made by the forget gate and the input gate layers. This adjustment involves point-wise multiplication and addition operations.As seen in equation (1d), f_t is used for point-wise multiplication with C_{t-1} to forget information from the context vector, while $i_t * \widetilde{C}_t$ represents new candidate values scaled by the extent to which they need to be updated. Ultimately, the process moves to the output state, where the sigmoid function determines which portion requires modification. This decision is then multiplied with the output obtained after the tanh that is used for scaling, resulting in hidden state output. A copy of h_t, denoted as y_t, serves as the predicted value.

IV. MATRIX PRODUCT STATE

Tensor Networks (TNs) are sparse data structures initially designed to efficiently simulate complex quantum systems in condensed matter, quantum chemistry, statistical mechanics and etc [9–11]. Conventional AI models are frequently perceived as "black box" tools, whereas TNs generative models offer a "white box" approach [12]. These models not only achieve performance comparable to the best available models but also provide unparalleled interpretability. MPS is a specific type of tensor network in which a high-order tensor is decomposed into a chain of low-order tensors connected sequentially. The MPS sets a new standard for transparency by providing clear, interpretable probabilities through its functions and tensors, rather than opaque weights. This advancement is crucial for tasks requiring a high level of trust and comprehension in AI decision-making, particularly in fields like finance, healthcare, and law. Recently, Stoudenmire and Schwab adapted tensor networks to supervised learning tasks, utilizing MPS to parameterize models for image classification. Their approach achieved impressive results, with less than 1% test set classification error [13]. There's also been a performance comparison between classical TNs and quantum approaches for image processing. Araz and Spannowsky concluded that classical TNs require exponentially large bond dimensions and higher Hilbert-space mapping to

perform comparably to their quantum counterparts. As dimensionality increases, classical TNs lead to a highly flat loss landscape, making the usage of gradient-based optimization methods challenging. They further investigated using Fisher information and effective dimensions, finding that classical TNs require a more extensive training sample to represent data as efficiently as TN-inspired quantum circuits [14]. Conversely, Laskaris et al. found that variational quantum circuits (VQC) approaches demonstrate advantages in terms of speed and accuracy when dealing with data characterized by a small number of features. However, for high-dimensional data, tensor networks surpass VQC in overall classification accuracy [15]. In addition to their application in computer vision, Jahromi and Orus have integrated TNs into neural networks (NN) to address scaling issues when NN dealing with a large number of neurons, which can limit the accessible number of layers [16]. Their study revealed a scalable tensor neural network (TNN) architecture capable of efficient training with a large number of neurons and layers. Their research demonstrates that TNNs can perform efficiently and accurately across various machine learning tasks. The inspiring research mentioned above has motivated us to incorporate MPS into LSTM networks to explore the feasibility of this fusion method for regression prediction. However, it's worth noting that there are numerous other tensor network architectures designed to address various types of problems. One well-known class is projected entangled pair states (PEPS), which represent a generalization of MPS to higher spatial dimensions [17, 18]. Another prominent and extensively utilized tensor network type is the multi-scale entanglement renormalization ansatz (MERA), constructed through successive coarse-graining applications [19]. Unlike MPS, MERA is capable of modeling critical behavior in one dimension and capturing the resultant power law decay of correlations. A variation of MERA is the tree tensor network (TTN), which lacks the unitary disentangler operations present in MERA [20].

V. INPUT DATA ENCODING

TN offer a means of exponentially compressing the state space, while still retaining accessibility to all pertinent physical features. Suppose $\mathbf{X}$ represents an input vector that undergoes transformation into a higher-dimensional space via a feature map $\Phi(\mathbf{X})$ before being evaluated by a decision function

$$f(\mathbf{X}) = W \cdot \Phi(\mathbf{X}). \tag{2}$$

where the feature vector $\Phi(\mathbf{X})$ and W can be exponentially large or even infinite. An established method for handling such large vectors is the widely recognized kernel trick that only requires working with scalar products of feature vectors, letting these vectors to be defined only implicitly [21]. Indeed, many successful applications of MPS or tensor train decomposition (see Figure 2) have been in quantum physics, where combining N independent systems requires taking the tensor product of their individual state vectors. By representing the weights W of Equation (2) as a MPS, we can efficiently optimize these weights and dynamically adjust their quantity by locally modifying a few tensors at a time. This process bears a striking resemblance to the density matrix renormalization group algorithm employed in physics. With the objective of leveraging similar TN concepts in machine learning, we opt for a feature map of the following structure:

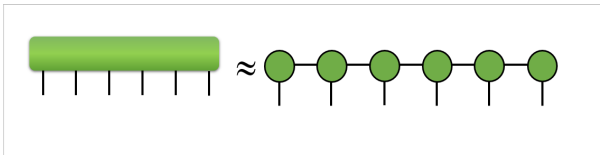
$$\Phi^{s_1 s_2 \cdots s_N}(\mathbf{X}) = \phi^{s_1}(x_1) \otimes \phi^{s_2}(x_2) \otimes \cdots \phi^{s_N}(x_N) \tag{3}$$

The tensor $\Phi^{s_1 s_2 \cdots s_N}$ represents the tensor product of the same local feature map $\phi^{s_k}(x_k)$, where the indices s_k range from 1 to d, where d being the local dimension. Consequently, each x_k is mapped to a d-dimensional vector, which must be ensured it has unit norm. This condition implies that each $\phi(x)$ also has unit norm.The complete feature map $\phi(x)$ can be deemed as a vector in a d^N-dimensional space or as an order-N tensor. The structure of $\phi(x)$ is illustrated in Figure 2b. Such a tensor is termed as rank-1 since it clearly emerges as the product of N order-1 tensors. From the physics point of view, $\phi(x)$ exhibits the same structure as a product state or unentangled wave function. While MPS are most effective for describing one-dimensional systems, their versatility allows them to be applied to higher-dimensional systems as well.

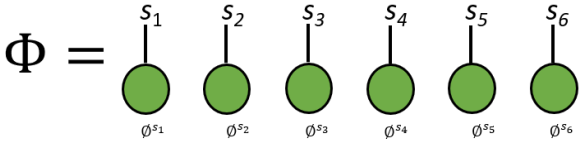
Our research involves processing a dataset consisting of 120 time-steps and F features, resulting in a total of N=120F inputs. Each element within this vector is encoded into a 2^N-dimensional vector space which means $d = 2$ in this study. Notably, every element undergoes encoding into its unique two-dimensional space via a local feature map. There are two potential options for this local feature map. The first is

$$\phi^{s_k}(x_k) = \left[\cos\left(\frac{\pi}{2}x_k\right), \sin\left(\frac{\pi}{2}x_k\right),\right] \tag{4}$$

where ϕ^{s_k} is the normalized wavefunction of a single qubit that can be conceptualized as representing spin up and down states within a quantum system. Another feature map that



(a) An example of a matrix product state (MPS) decomposition, where an order-N tensor is approximated by a contracted chain of N lower-order tensors. Lines represent tensor indices, and connecting two lines denotes summation.



(b) An example of input data being mapped into a normalized order-N tensor with a ranked-1 product structure.

FIG. 2: Graphical tensor description for Matrix Product State.

has the same properties can be written as

$$\phi^{s_k}(x_k) = [1 - x_k, x_k] \tag{5}$$

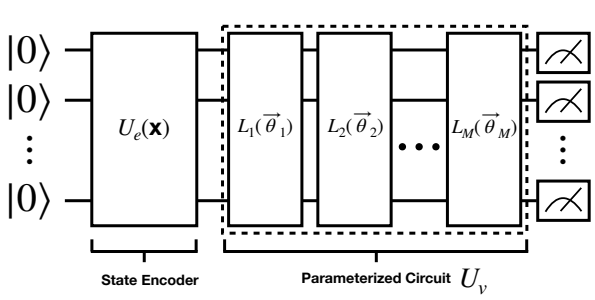
The entire dataset of size N is represented as the tensor product of these individual local vectors. We may refer to this object as a "*data tensor*" or "*data state*" [22]. The MPS tensor network represents yet another type of vector within the seismic data space, comprising 2^N components. Each component within the MPS can be expressed as the product of N matrices.Thus, an MPS decomposition of the weight tensor W has the form

$$W_{s_1 s_2 \cdots s_N} = \sum_{\{\alpha\}} A_{s_1}^{\alpha_1} A_{s_2}^{\alpha_1 \alpha_2} \cdots A_{s_k}^{\alpha_k \alpha_{k+1}} \cdots A_{s_N}^{\alpha_{N-1}}. \tag{6}$$

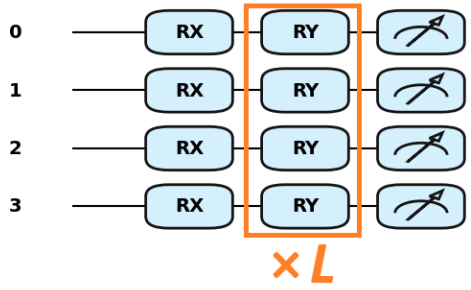
The ranges of each of the α_k indices, termed as bond dimensions, serve as hyperparameters for the model. These bond dimensions dictate the sizes of the tensors. The elements of the tensors act as variational parameters or weights, which are fixed through training. There exists some redundancy in these parameters, referred to as gauge freedom, but we will not delve into that aspect here [23]. In typical physics applications, the MPS bond dimension m can vary from 10 to 10,000 or even higher. For the most complex physics systems, it's desirable to allow for as large a bond dimension as possible because a larger dimension typically translates to higher accuracy. However, when employing MPS in a machine learning

context, the bond dimension directly influences the number of model parameters. Unlike in physics, opting for an excessively large bond dimension might not be advantageous as it could result in over-fitting.

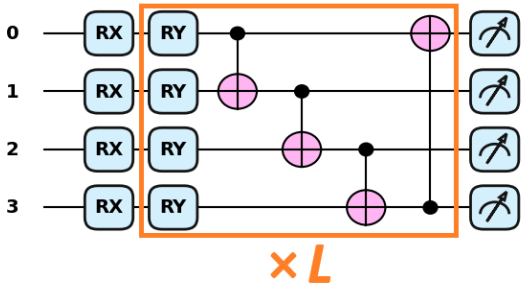
The dataset contains a total of 44 features. After examining its characteristics through covariance matrix analysis, SHAP analysis and preliminary study, we removed date-related features such as Lstd, Hn, Hnmax, and others, as highlighted in Table I in bold. In total, 19 features were used.



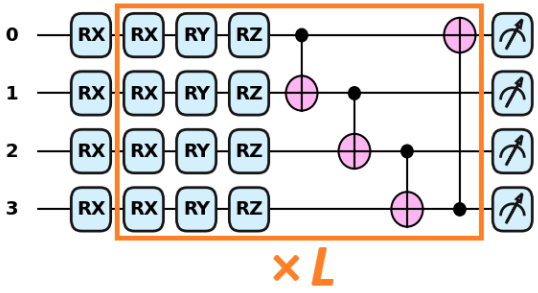
(a) Generic architecture for variational quantum circuits (VQC).



(b) Simple L-Layer ansatz using only R_y rotations.



(c) Simple L-Layer ansatz with R_y rotations and entanglement



(d) Complex L-Layer ansatz with R_x, R_y, R_z rotations and entanglement.

FIG. 3: Examples of VQC designs.

VI. VARIATIONAL QUANTUM CIRCUITS

Variational Quantum Circuits (VQCs), also referred to as Parameterized Quantum Circuits (PQCs), are composed of tunable parameters that can be iteratively optimized. The term variational indicates that certain parameters within the circuit are updated according

to a predefined loss function during the optimization process. The computation of the loss function is handled by classical computing resources, while the parameters themselves are optimized using classical machine learning techniques, including both gradient-based and non-gradient-based algorithms. As illustrated in Figure (3a) , the Variational Quantum Circuit (VQC) architecture consists of three main components. The first is the state encoding module $U_e(\mathbf{x})$, which maps classical input data $\mathbf{x}$ into quantum information by preparing qubit states suitable for quantum processing. The second is the variational module U_v, which entangles qubit states using controlled gates (e.g., CNOT) and applies parameterized rotations (e.g. R_x, R_y and R_z). To enhance the performance of the circuit, the option to construct a multi-layer/multi-block structure is permitted, enabling the incorporation of additional learning parameters. To improve circuit expressivity, this module may be extended into a multi-layer or multi-block structure, thereby introducing additional trainable parameters. Its overall operation can be expressed as $U_v(\Theta) = L_M(\vec{\theta}_M)L_{M-1}(\vec{\theta}_{M-1})\cdots L_1(\vec{\theta}_1)$, where M denotes the total number of layers and $\Theta = \{\vec{\theta_1}, \vec{\theta_2}\cdots\vec{\theta_M}\}$. represents the complete set of trainable parameters. The final component is the measurement module $\mathcal{M}$, which extracts expectation values from the qubit states. The circuit is executed multiple times (known as shots) to estimate these expectations, with the Pauli-Z operator being a common choice in quantum machine learning applications. A key advantage of VQCs is their ability to integrate seamlessly with classical frameworks such as tensor networks and deep neural networks [24]. This flexibility enables both pre-processing (e.g., dimensionality reduction via circuit dressing [25]) and post-processing to achieve scaling objectives. The overall computational process within the VQC framework can be expressed as:

$$f(x,\theta) = \bigotimes \Pi_{M\in\mathcal{M}} \langle 0| U_e^\dagger(x)U_v^\dagger(\theta)MU_v(\theta)U_e(x)|0\rangle , \tag{7}$$

where $\bigotimes$ denotes the qubit superposition operator, while $f(x,\theta)$ epresents the output of the VQC given inputs x and parameters θ. The set $\mathcal{M}$ corresponds to the quantum measurement bases in the VQC, with its cardinality bounded above by the number of qubits. The operation of the VQC employed in this work is given as $\overrightarrow{f(\mathbf{x};\Theta)} = \left(\left\langle \hat{Z}_1 \right\rangle, \cdots, \left\langle \hat{Z}_q \right\rangle\right)$, where $\left\langle \hat{Z}_k \right\rangle = \left\langle 0 \left| U_e^\dagger(\mathbf{x})U_v^\dagger(\Theta)\hat{Z}_k U_v(\Theta)U_e(\mathbf{x}) \right| 0 \right\rangle$ and q represents the total number of qubits measured in the system. The expectation values $\left\langle \hat{Z}_k \right\rangle$ can be computed analytically when the circuit is simulated classically. During simulation, the quantum circuit parameters are updated at each training epoch, with the optimizer guiding the parameter search toward

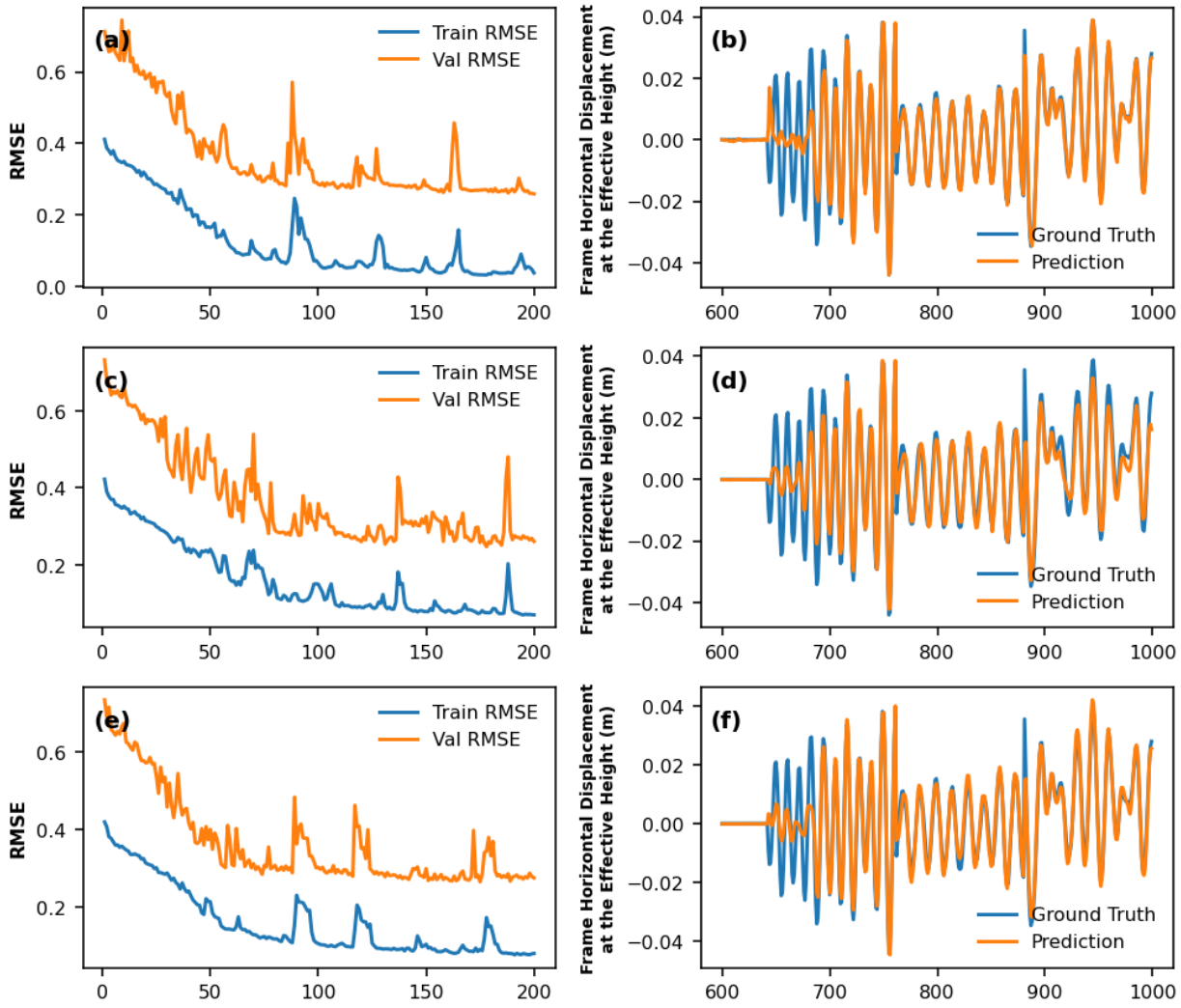


FIG. 4: Results for 3-layer LSTM with 192 hidden size. Subfigures (a), (c), and (e) show the RMSE values for 36 input features with dropout rates of 0.0, 0.25, and 0.30, respectively. Subfigures (b), (d), and (f) present the prediction results compared against the ground truth.

optimal values that align with the target outputs.

VII. RESULT DISCUSSION

The dataset employed in this study was provided by the authors, as referenced in [26]. The complete set of features is listed in Table I. As previously explained, the dataset was constructed to ensure reliability by enforcing event-wise data integrity, adopting non-overlapping

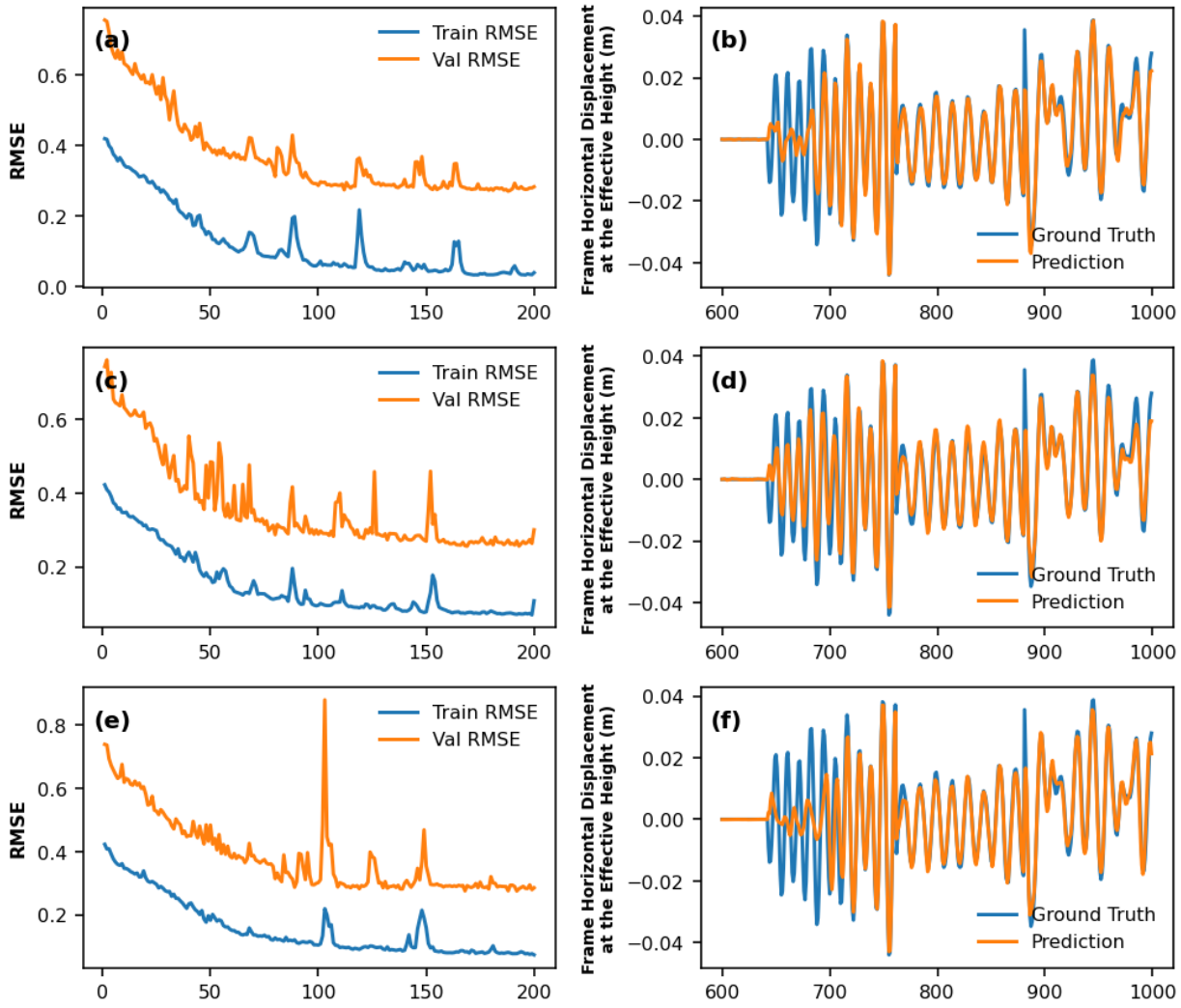


FIG. 5: Results for 3-layer LSTM with 192 hidden size. Subfigures (a), (c), and (e) show the RMSE values for 19 input features with dropout rates of 0.0, 0.25, and 0.30, respectively. Subfigures (b), (d), and (f) present the prediction results compared against the ground truth.

sequence construction, and guaranteeing that no causal or material continuity exists between distinct seismic events. This design supports sequence-to-sequence prediction while avoiding physically invalid outcomes for the trained model. The dataset is systematically organized using unique identifiers, specifically the combination of $AnalysisNo, Frame\_Name$, and EQ_Record_Name, ensuring that each group corresponds to a single, self-contained dynamic experiment. From an initial pool of over one million records, approximately 100K

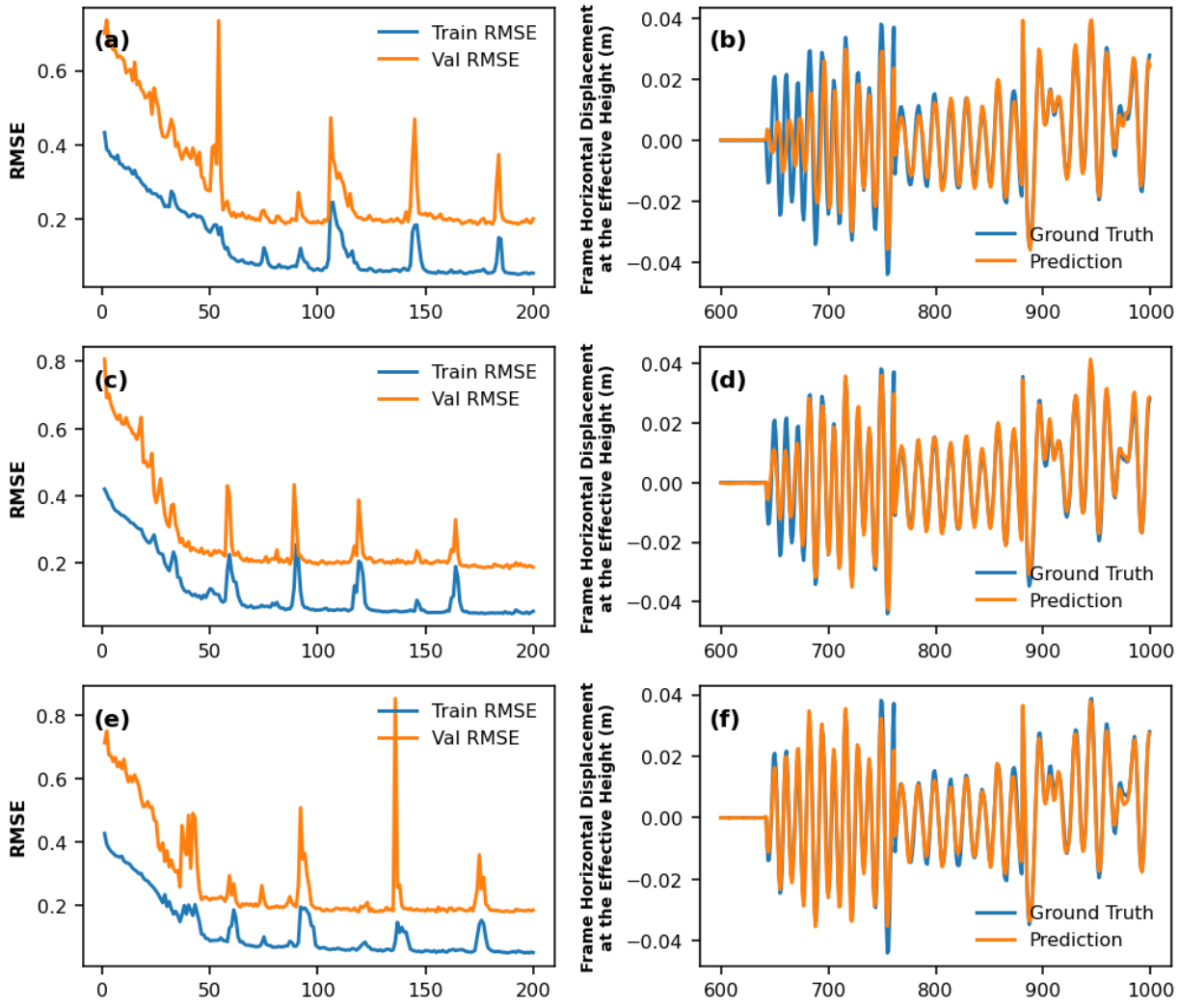


FIG. 6: Results for the 3-layer Bi-LSTM model with a hidden size of 192. Subfigure (a) reports the RMSE obtained using 36 input features with a dropout rate of 0.25. Subfigures (c) and (e) show RMSE results with 19 input features, using dropout rates of 0.25 and 0.24, respectively. Subfigures (b), (d), and (f) illustrate the corresponding prediction outputs compared against the ground truth.

rows were selected to form the working dataset. Based on these, 842 sequences were constructed, with 589 used for training, 168 for validation, and the remainder for testing, resulting in a $70 : 20 : 10$ distribution in this study. It should be noted that the entire training process was implemented in PyTorch, applying the min-max normalization, utilizing the AdamW optimizer with a learning rate of 0.001, batch size of 4, weight decay of 0.0001,

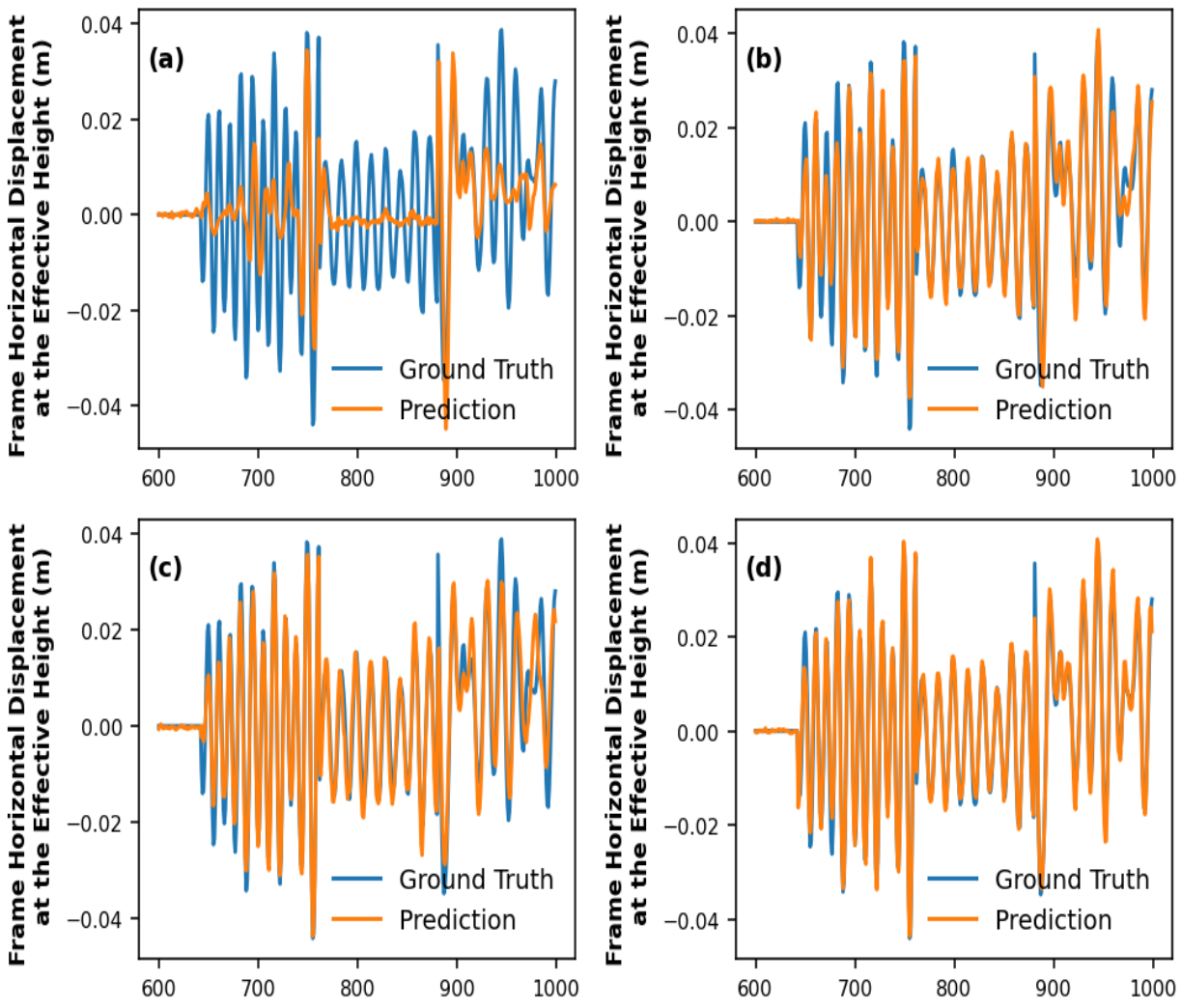


FIG. 7: Results for the MPS + 2-layer LSTM model with hidden size 192, dropout rate 0.20, bond size 9, and 19 input features. Subfigures (a)–(d) present prediction outputs compared with the ground truth for MPS output features of 10, 12, 14, and 16, respectively.

and a CosineAnnealingLR scheduler that further smooths optimization dynamics over long epochs, improving convergence robustness. Note that for the inference process, we employed the model that achieved the lowest validation RMSE, as saved during the 200-epoch training. It is to emphasis that the solution uses LSTM as the core for regression, with memory reset at each event. Since episodes are independent sequences, hidden states are cleared at sequence boundaries to prevent carryover across events. During early experimentation,

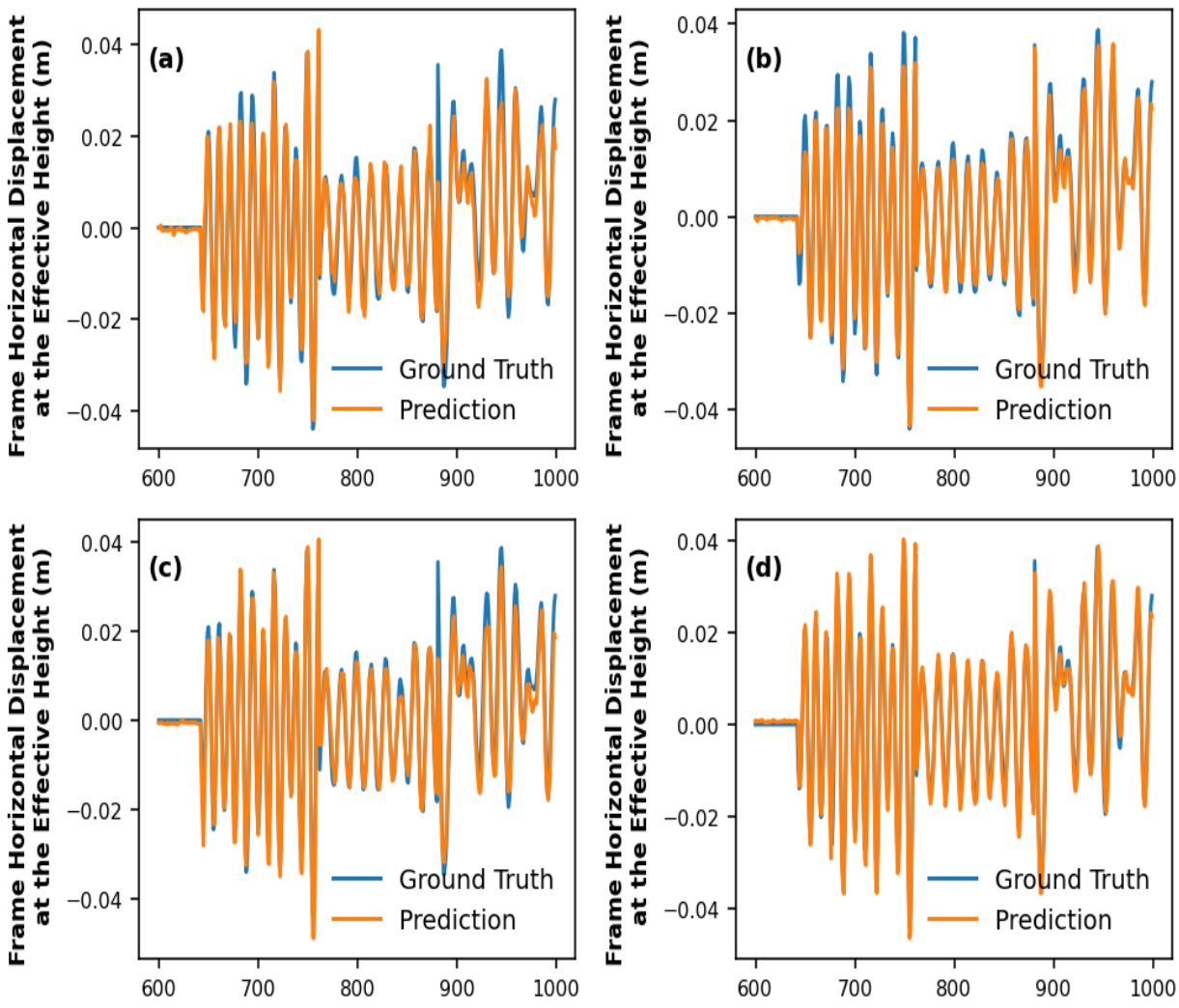


FIG. 8: Results for the MPS + 2-layer LSTM model with hidden size 192, dropout rate 0.20, 19 input features and 14 output features. Subfigures (a)–(d) show prediction outputs compared with the ground truth for MPS bond sizes of 6, 8, 10, and 12, respectively.

occasional RMSE spikes were observed at random batches. These were traced to "context switching" between distinct structural–seismic regimes when the model attempted to process unrelated event sequences within the same batch.

A. Train a purely classical LSTM model

In this section, we examine the expressivity of the classical LSTM model using all features that are not directly related to the target variables, such as $ADE\_frame\_disp\_max\_sofar\_m$, $ADE\_frame\_$

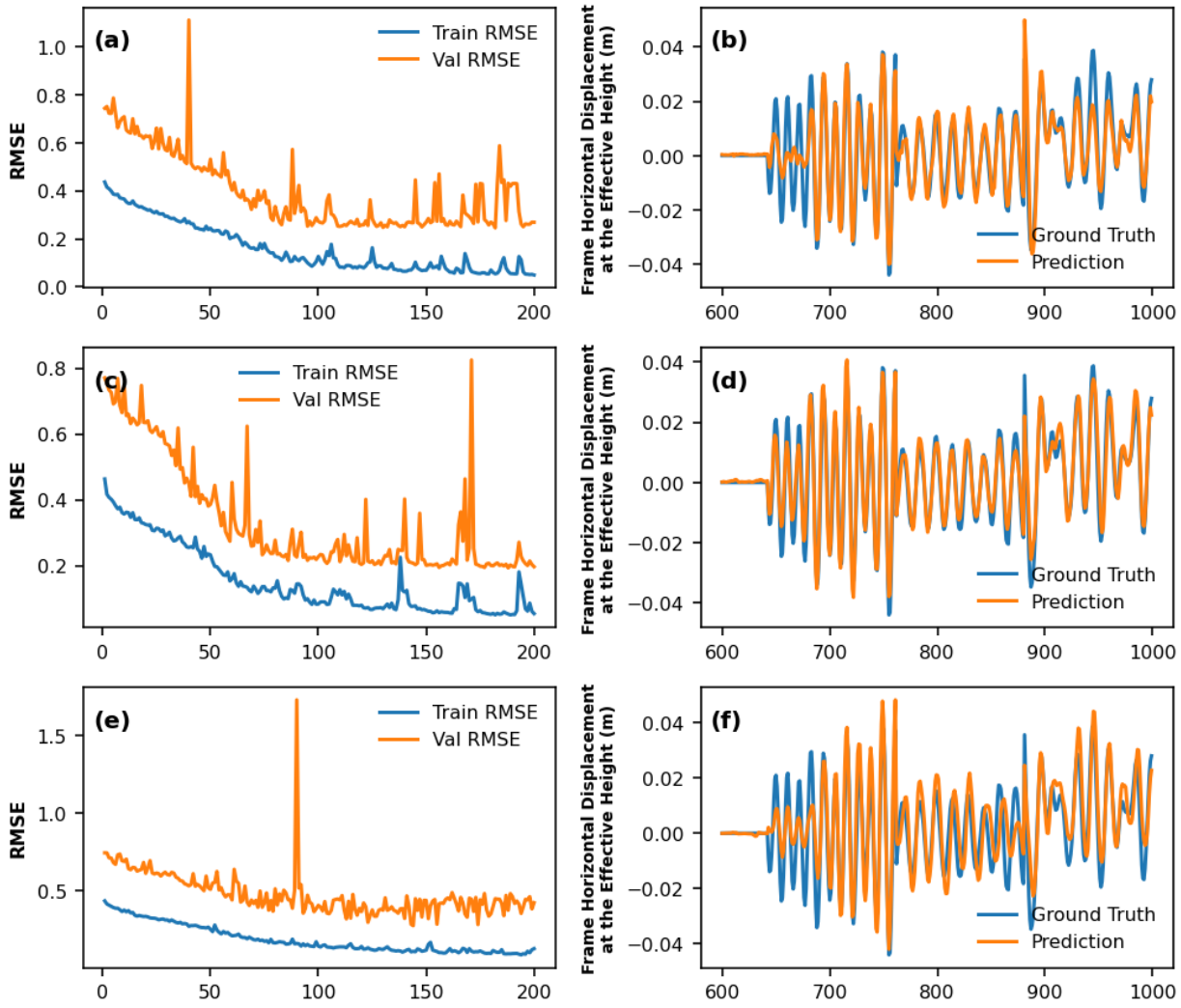


FIG. 9: Results obtained with TorchQuantum for the CNN (32 channels) combined with a 3-layer QLSTM and a 2-layer LSTM (hidden size 96, 5 qubits, 19 input features). Subfigures (a), (c), and (e) present RMSE results with dropout rates of 0.00, 0.10, and 0.20, respectively. Subfigures (b), (d), and (f) show the corresponding prediction outputs compared against the ground truth.

and $ADE\_frame\_1steps\_back\_disp\_m$. As shown in Figure (4), the training and validation normalized loss curves in seismic modeling often exhibit noisy fluctuations. These instabilities reflect the transition between batches representing different physical events whose response characteristics vary substantially, resulting in nonuniform gradient dynamics during training. Nevertheless, no overfitting or underfitting conditions are observed. Referring

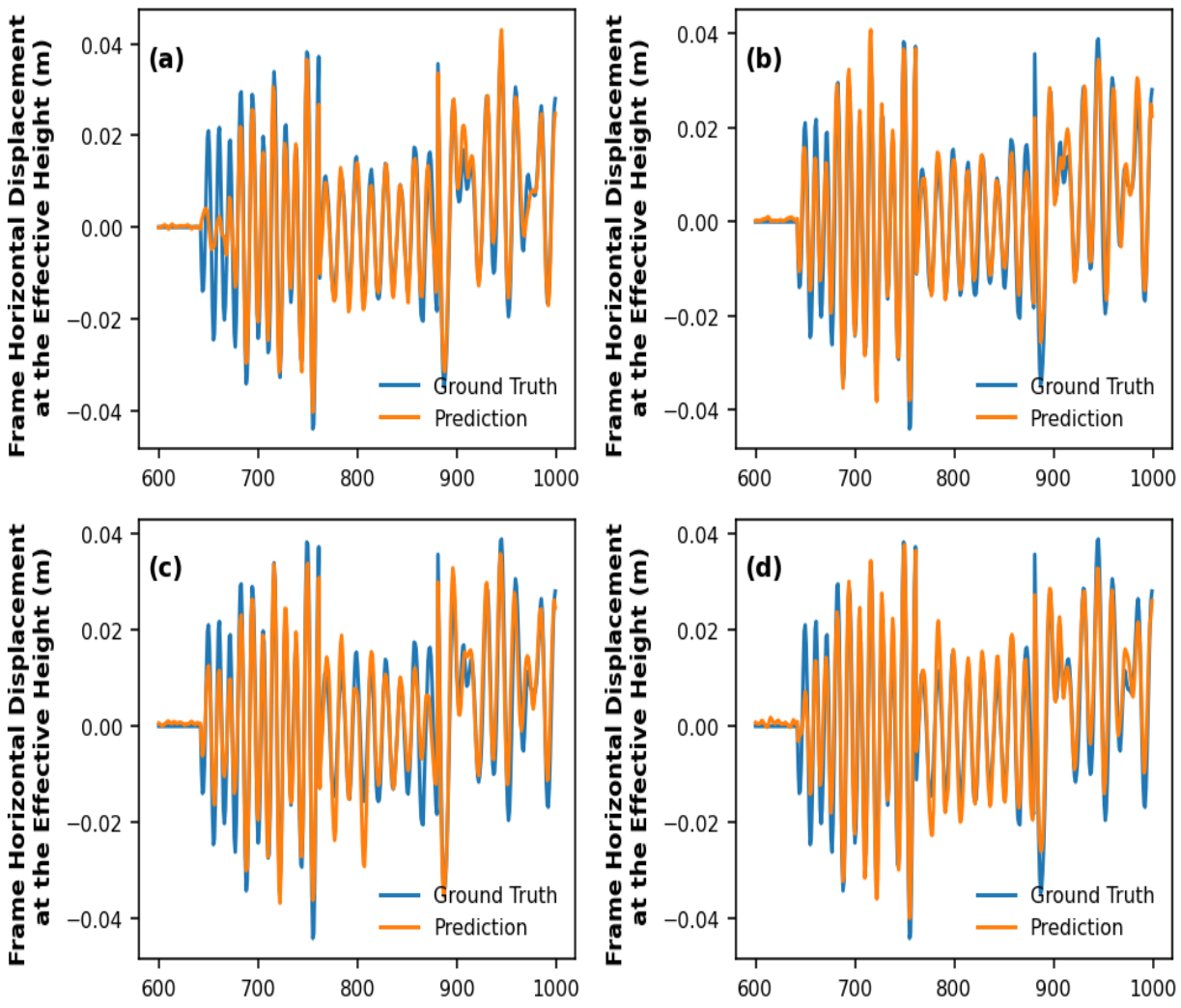


FIG. 10: Results obtained using TorchQuantum for the CNN (32 channels) + 3-layer QLSTM + 2-layer LSTM model with hidden size 96, dropout rate 0.10, and 19 input features. Subfigures (a)–(d) display prediction outputs compared with the ground truth for qubit counts of 4, 5, 6, and 7, respectively.

to Table II, the real test RMSE values converge to 0.057, 0.0052, and 0.0057 for dropout rates of 0.00, 0.25, and 0.30, respectively. All three configurations yield weak predictions within the testing range of 650–700. Among them, the model with a dropout rate of 0.25 performs best when evaluated using RMSE and R^2 metrics. With the number of features reduced to 19, as shown in Figure (5), the RMSE curves become more stable after approximately 160 epochs. A clear comparison between predicted values and ground truth

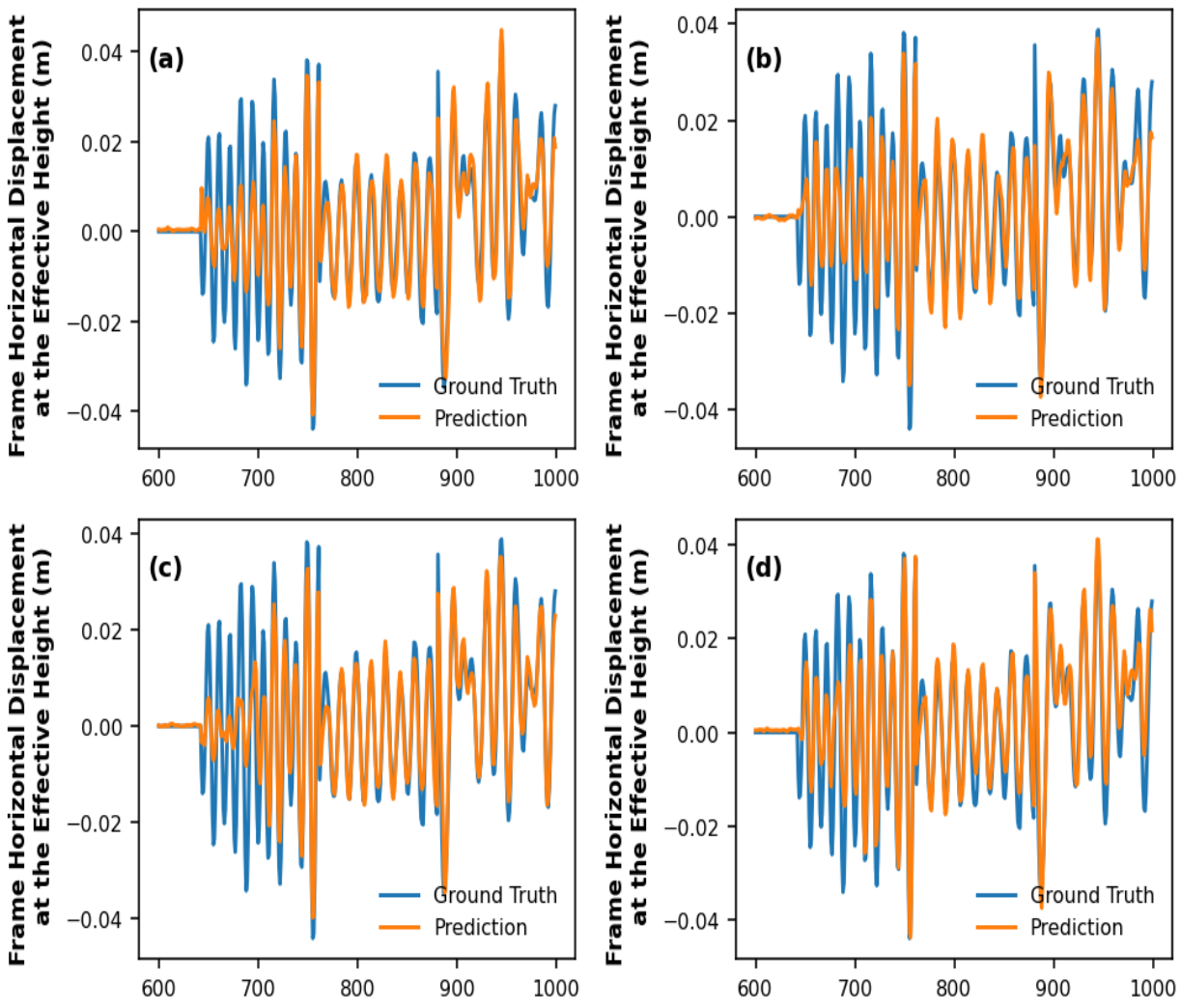


FIG. 11: Results obtained with TorchQuantum for the CNN (32 channels) combined with a 3-layer QLSTM and a 2-layer LSTM (hidden size 96, 5 qubits, dropout rate 0.10, 19 input features). Subfigures (a)–(d) show prediction outputs compared against the ground truth for different VQC architecture designs.

indicates that Figure (5d), corresponding to a dropout rate of 0.25, provides the best match. The associated lowest real test RMSE and R^2 values are 0.0048 and 0.9387, respectively. We further investigate a model trained using a variant of LSTM, namely the bidirectional LSTM (Bi-LSTM). This architecture consists of a forward module that reads the sequence from start to end and a backward module that reads the sequence from end to start. The outputs from both directions are combined to form the final representation at each time

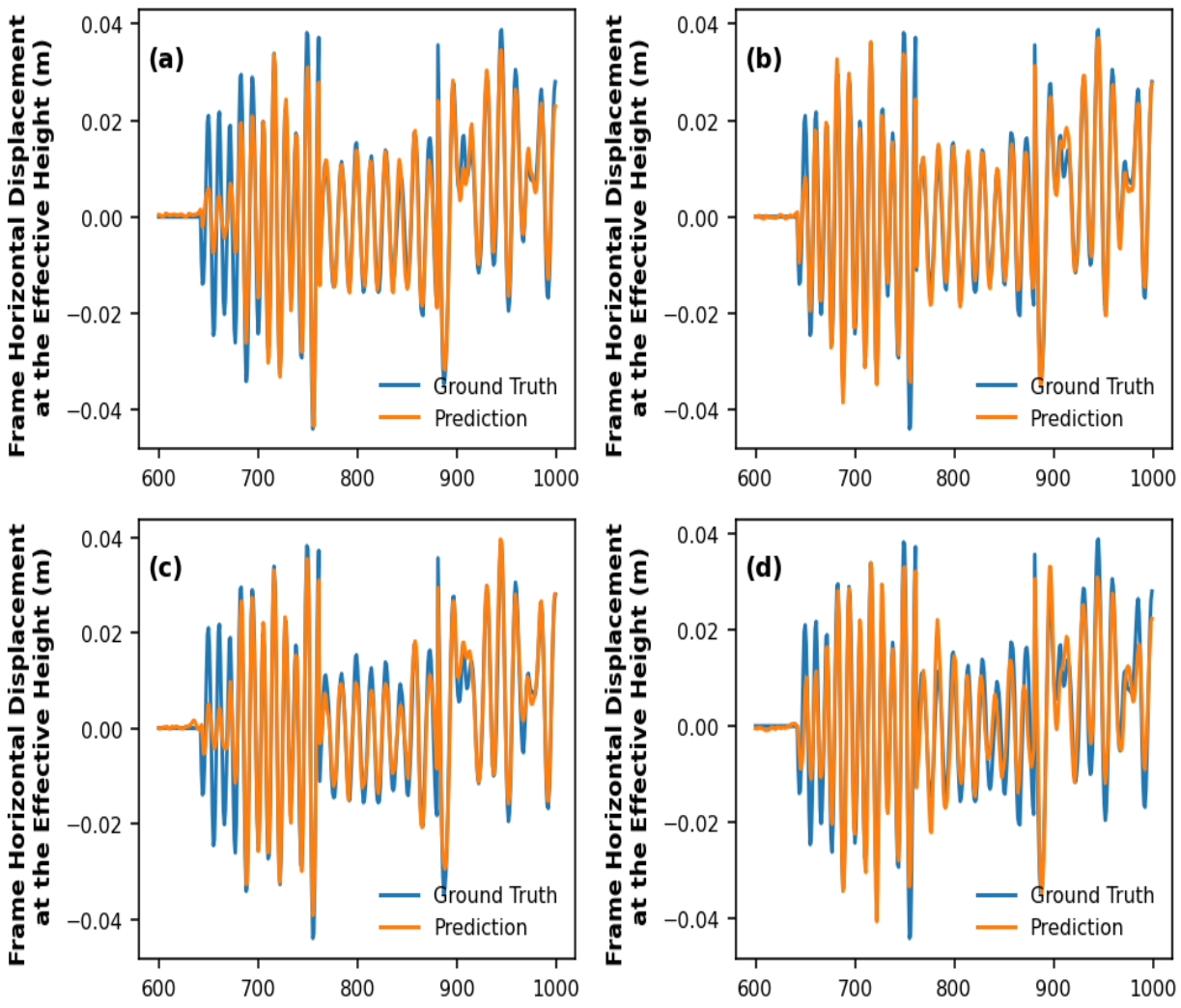


FIG. 12: Results for the CNN (32 channels) combined with a 3-layer QLSTM and a 2-layer LSTM (hidden size 96, 5 qubits, dropout rate 0.10, 19 input features). Subfigures (a) and (c) present results obtained without entanglement. The figures compare prediction outputs against the ground truth using Pennylane and Qrisp.

step, thereby enhancing the expressivity of the trained model. As depicted in Figure (6), although the normalized RMSE curves exhibit several peaks, they show fewer fluctuations compared to the standard LSTM. With 36 features, as given in Figure (6b), the Bi-LSTM does not outperform the model trained with a selected set of 16 input features, regardless of whether the dropout rate is 0.24 or 0.25. According to Table II, the smallest real test RMSE values are 0.0069, 0.0031, and 0.0028, while the corresponding R^2 values are 0.08739

and 0.9750. These results indicate that the model with 16 input features and a dropout rate of 0.24 provides the most accurate prediction. However, implementing Bi-LSTM comes at the cost of increased energy consumption, as the total number of trainable parameters in this model is approximately 2.8 times greater than that of a standard LSTM.

B. Hybrid Modeling: MPS-Enhanced Classical LSTM

Tensor networks offer a structured way to represent and process multi-dimensional data, enabling more efficient learning algorithms. By incorporating MPS into LSTM that capable of compressing high-dimensional inputs into efficient tensor representations, reducing parameters while enhancing expressivity. Thus, this hybrid model is particularly valuable in domains like seismic modeling, where large feature sets must be processed without sacrificing accuracy. In this model, additional hyper-parameters are introduced, namely the bond dimension and the number of output features. The bond dimension refers to the size of the index connecting neighboring tensors in the chain, governing the trade-off between compression efficiency and expressive power. The output features correspond to the contracted tensor (or the final site in the chain) that produces a compressed feature vector. We first fix the bond dimension at 9 and vary the output features to 10, 12, 14, and 16. As shown in Figure (7), compressing the features to 10 results in a substantial loss of useful information, hindering the training of a robust model. When the number of output features exceeds 12, the predicted values align more closely with the ground truth. However, it is difficult to determine the best choice among 12, 14, and 16 output features. Referring to Table II, the configuration with 16 output features achieves the lowest test RMSE (0.0031) and the highest R^2 (0.9747). Interestingly, the case with 14 output features performs worse than that with 12, based on these two metrics. Therefore, increasing the number of output features does not necessarily lead to a more accurate model in this hybrid architecture. Compared to a model that relies solely on the classical LSTM architecture, the MPS-LSTM hybrid significantly reduces the number of trainable parameters. For example, the model in Figure (5) requires 769,729 parameters, whereas the hybrid model with 16 output features requires only 482,705 parameters, representing a savings of approximately 38% in computational resources.

The following study examines the impact of bond dimension while keeping the output

features fixed at 14. As shown in Figure (8), varying the bond dimension from 6 to 12 yields predictions that align well with the ground truth, with the case of bond dimension 12 nearly overlapping the blue reference line. However, Table II reveals a different perspective: the model with bond dimension 8 offers slight advantages in terms of R^2 and SMAPE. In addition, the bond dimension 8 configuration requires 466,059 trainable parameters, compared to 494,871 for bond dimension 10, thereby reducing the parameter count by 28,812. Taken together, these results indicate that the model with bond dimension 8 provides the most efficient and accurate choice.

C. Hybrid Architecture: CNN–QLSTM–LSTM Integration

This architecture employs a CNN to extract local temporal features while reducing sequence length. In contrast, the QLSTM block processes a compressed global representation through quantum gates, enabling richer feature encoding. Finally, the classical Bi-LSTM integrates the CNN-derived features with quantum outputs across time to perform sequential prediction. In summary, the hybrid design combines CNN for locality, QLSTM for quantum-enhanced global encoding, and Bi-LSTM for temporal dynamics, yielding a principled framework for time-series and sequential tasks where both fine-grained detail and global structure are essential.

In Figure (9), we examine the effect of dropout rate on the normalized training and validation RMSE. A dropout rate of 0.20 acts as a regularizer, mitigating overfitting to noise or spurious patterns. Consequently, the RMSE curve exhibits reduced variance across epochs, appearing smoother with fewer sharp peaks or fluctuations. However, this does not necessarily yield lower RMSE values, as shown in Figure (9e), where the RMSE is noticeably higher than in Figures (9a) and (9c). In terms of predictive accuracy, a dropout rate of 0.10 produces results that align more closely with the ground truth. The corresponding R^2 values for dropout rates of 0.0, 0.10, and 0.20 under the 32-channel CNN, Bi-LSTM, and 5-qubit QLSTM (forget, update, input, and output gates use Figure 3(b) design) configuration are 0.8933, 0.9487, and 0.7491, respectively. These findings indicate that a dropout rate of 0.10 is the most suitable choice for this system. The subsequent stage investigates the appropriate number of qubits for the hybrid model. Figure (10) illustrates results for qubit counts of 4, 5, 6, and 7, showing that 5 qubits are sufficient to train a robust regression

model. Nonetheless, across all cases, predictions in the range of 650–700 do not fully align with the ground truth. With 5 qubits, the hybrid model achieves a test RMSE of 0.0044 and R^2=0.9487, representing the best performance among the tested configurations.

D. VQC Architectures and Quantum Frameworks in CNN–QLSTM–LSTM

In Figure (11), we compare different ansatz designs for the VQC within the QLSTM. Figure (11a) employs a simple entangled circuit, as shown in Figure (3c), with three quantum layers applied to the forget, update, input, and output gates; this configuration is referred to as the Simple-ET architecture. In Figure (11b), the forget and output gates adopt a more complex design involving Rx, Ry, and Rz rotations with entanglement, illustrated in Figure (3d), and termed MIX1. Figure (11c) exchanges the circuit assignments, applying the design from Figure (3b) to the forget and output gates, yielding the MIX2 architecture. Figure (11d) applies the design of Figure (3d) uniformly across all gates, requiring numerous rotation operations and resulting in 120 quantum trainable parameters; this is denoted as CPLX. Visual inspection suggests that Figures (11b) and (11d) align more closely with the ground truth, though it is difficult to determine the superior model. Table II provides further clarity: the (test RMSE, R^2) values for Simple-ET, MIX1, and CPLX are (0.0065, 0.8891), (0.0066, 0.8852), and (0.0058, 0.9102), respectively. These results indicate that CPLX achieves the best performance, likely due to its larger number of quantum trainable parameters, which capture more informative correlations. Interestingly, Simple-ET, despite having fewer parameters, outperforms both MIX1 and MIX2, suggesting that the QLSTM design may be more efficient when paired with simpler VQC structures. The comparison between MIX1 and MIX2 further implies that a straightforward VQC design for the forget and output gates yields better predictive accuracy. Numerous quantum frameworks have been developed to date. In this study, we compare three frameworks applied to the same QLSTM architecture: TorchQuantum, PennyLane, and Qrisp. Figures (12a and (12b) present models trained with PennyLane, with and without entanglement, while Figures (12c) and (12d) show results obtained using Qrisp under the same conditions. Within the dataset range of 650–700, both frameworks consistently demonstrate that predictions without entanglement are less accurate, and performance improves when entanglement is introduced. Table II further quantifies this effect: PennyLane achieves (0.0037, 0.9644) with entanglement

and (0.0047, 0.9409) without, whereas Qrisp yields (0.0054, 0.9236) with entanglement and (0.0045, 0.9460) without. Interestingly, Qrisp produces a model where the non-entangled configuration outperforms the entangled one. Revisiting TorchQuantum, the non-entangled results are (0.0044, 0.9487), indicating that across all three frameworks, performance is broadly comparable.

VIII. CONCLUSION

In this work, we employ classical,quantum-inspired model MPS+LSTM, and hybrid classical quantum LSTM to predict the full time-history response of structures, leveraging carefully preprocessed and cleaned seismic datasets with non-overlapping sequences to ensure robust training. Unlike prior approaches, our models do not require historical response data, yet produce accurate regression predictors. These findings indicate that classical LSTM architectures already achieve efficient training on modern GPUs, while quantum-inspired models such as MPS-LSTM further enhance representational compactness. In addition, hybrid classical–quantum architectures like QLSTM exhibit promising potential, and future fully quantum computing implementations may deliver additional acceleration in both training and inference, enabling scalable and accurate seismic vulnerability assessment for individual structures. Future possible research may consider integration of physics-informed neural networks with QLSTM that could embed structural dynamics directly into the learning process, improving interpretability and robustness for seismic vulnerability assessment.

ACKNOWLEDGMENTS

Supported by EU Horizon projects OASEES (ID:101092702) and SecQDevOps (ID:101225776). Technical support from Dr. Andrew Beng Jin Teoh (Yonsei University) is gratefully acknowledged. We would also like to thank both Prof. Ihsan Engin and Eleni Smyrou (Hanze University of Applied Sciences), and Mesut Boğaç Kaya (SENSO Engineering BV) for their

technical assistance, which contributed to improving the overall quality of this work.

---

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

[2] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *European Journal of Operational Research*, vol. 270, no. 2, pp. 654–669, 2018.

[3] E. Chatzi, T. Simpson, and N. Dervilis, "Machine learning approach to model order reduction of nonlinear systems via autoencoder and lstm networks," *Journal of Engineering Mechanics*, 2021.

[4] S. Chien, D. Chieng, S. Y. Chen, C. C. Zarakovitis, H. S. Lim, and Y. Xu, "Applying hybrid quantum lstm for indoor localization based on rssi," in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 131–135, 2024.

[5] R. Yu, S. Zheng, A. Anandkumar, and Y. Yue, "Long-term forecasting using higher order tensor rnns," *arXiv: Learning*, 2017.

[6] J. B. Moore, H. P. Stackhouse, B. D. Fulcher, and S. Mahmoodian, "Using matrix-product states for time-series machine learning," *Physical Review Research*, vol. 7, no. 4, p. 043010, 2025.

[7] I. E. Bal and E. Smyrou, "Implementation of emerging technologies in seismic risk estimation," in *Progresses in European Earthquake Engineering and Seismology* (R. Vacareanu and C. Ionescu, eds.), (Cham), pp. 279–294, Springer International Publishing, 2022.

[8] H. J. Jang, J.-M. Shin, and L. Choi, "Geomagnetic field based indoor localization using recurrent neural networks," *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pp. 1–6, 2017.

[9] D. Perez-Garcia, F. Verstraete, M. M. Wolf, and J. I. Cirac, "Matrix product state representations," *arXiv preprint quant-ph/0608197*, 2006.

[10] S. Szalay, M. Pfeffer, V. Murg, G. Barcza, F. Verstraete, R. Schneider, and Ö. Legeza, "Tensor product methods and entanglement optimization for ab initio quantum chemistry," *International Journal of Quantum Chemistry*, vol. 115, no. 19, pp. 1342–1391, 2015.

[11] G. Evenbly and G. Vidal, "Tensor network renormalization," *Physical review letters*, vol. 115, no. 18, p. 180405, 2015.

[12] S.-J. Ran and G. Su, "Tensor networks for interpretable and efficient quantum-inspired machine learning," *Intelligent Computing*, vol. 2, p. 0061, 2023.

[13] E. Stoudenmire and D. J. Schwab, "Supervised learning with tensor networks," *Advances in neural information processing systems*, vol. 29, 2016.

[14] J. Y. Araz and M. Spannowsky, "Classical versus quantum: Comparing tensor-network-based quantum circuits on large hadron collider data," *Physical Review A*, vol. 106, no. 6, p. 062423, 2022.

[15] G. Laskaris, A. A. Melnikov, M. R. Perelshtein, R. Brasher, T. Baeck, and F. Neukart, "Comparison between tensor networks and variational quantum classifier," *arXiv preprint arXiv:2311.15663*, 2023.

[16] S. S. Jahromi and R. Orús, "Variational tensor neural networks for deep learning," *arXiv preprint arXiv:2211.14657*, 2022.

[17] F. Verstraete, M. M. Wolf, D. Perez-Garcia, and J. I. Cirac, "Criticality, the area law, and the computational power of projected entangled pair states," *Physical review letters*, vol. 96, no. 22, p. 220601, 2006.

[18] F. Verstraete and J. I. Cirac, "Renormalization algorithms for quantum-many body systems in two and higher dimensions," *arXiv preprint cond-mat/0407066*, 2004.

[19] G. Vidal, "Class of quantum many-body states that can be efficiently simulated," *Physical review letters*, vol. 101, no. 11, p. 110501, 2008.

[20] Y.-Y. Shi, L.-M. Duan, and G. Vidal, "Classical simulation of quantum many-body systems with a tree tensor network," *Physical review a*, vol. 74, no. 2, p. 022320, 2006.

[21] K.-R. Müller, S. Mika, K. Tsuda, and K. Schölkopf, "An introduction to kernel-based learning algorithms," pp. 4–1, 2018.

[22] S. Efthymiou, J. Hidary, and S. Leichenauer, "Tensornetwork for machine learning," *arXiv preprint arXiv:1906.06329*, 2019.

[23] G. Evenbly, "Gauge fixing, canonical forms, and optimal truncations in tensor networks with closed loops," *Physical Review B*, vol. 98, no. 8, p. 085155, 2018.

[24] S. Y.-C. Chen, C.-M. Huang, C.-W. Hsing, and Y.-J. Kao, "Hybrid quantum-classical classifier based on tensor network and variational quantum circuit," *arXiv preprint arXiv:2011.14651*,

2020.

[25] S. Y.-C. Chen, C.-H. H. Yang, J. Qi, P.-Y. Chen, X. Ma, and H.-S. Goan, "Variational quantum circuits for deep reinforcement learning," *IEEE access*, vol. 8, pp. 141007–141024, 2020.

[26] I. E. Bal and E. Smyrou, "Implementation of emerging technologies in seismic risk estimation," in *Progresses in European Earthquake Engineering and Seismology* (R. Vacareanu and C. Ionescu, eds.), (Cham), pp. 279–294, Springer International Publishing, 2022.

TABLE I: Features Before Cleansing

Number	Parameter	Description
1	ro	Column reinforcement ratio
2	fc	Concrete quality (MPa)
3	fs	Steel quality (MPa)
4	Lmin	Minimum span length (m)
5	**Lmax**	Maximum span length (m)
6	**Lmed**	Median span length (m)
7	**Lstd**	Standard deviation of span length (m)
8	Hs	Ground storey height (m)
9	HsHn	Ground-to-normal storey height ratio
10	**Hn**	Nominal height of a regular storey (m)
11	**Hnmax**	Maximum storey height in the structure
12	**hmin**	Min. column depth in frame direction (upper floors)
13	**hmax**	Max. column depth in frame direction (upper floors)
14	**hmed**	Median column depth in frame direction (upper floors)
15	**hstd**	Standard column depth (or cross-section) in the frame direction (m)
16	**hgmin**	Min. column depth in frame direction (ground floor)
17	**hgmax**	Max. column depth in frame direction (ground floor)
18	**hgmed**	Median column depth in frame direction (ground floor)
19	**Hb**	Base column depth or base width (m)
20	**Vs30**	Shear wave velocity of the site (m/s)
21	PGA_g	Peak ground acceleration (g)
22	PGV_cmsec	Peak ground velocity (cm/s)
23	Arias	Arias intensity (m/s)
24	t5_75	Duration from 5–75% (s)
25	Tm	Middle period(s)
26	**Sa_T02_g**	Spectral acceleration at $T = 0.2$ s (g)
27	**Sa_T10_g**	Spectral acceleration at $T = 1.0$ s (g)
28	Sd_T02_cm	Spectral displacement at $T = 0.2$ s (cm)
29	**Sd_T10_cm**	Spectral displacement at $T = 1.0$ s (cm)
30	Acc_gal	Ground acceleration (gal)
31	Acc_Avg_gal	Average absolute ground acceleration (gal)
32	Acc_Slope_gal	Acceleration slope (gal/s)
33	T1	Fundamental period of the structure (first-mode) (m)
34	**Deff**	Effective lateral displacement (m)
35	MaxDr	Maximum inter-storey drift ratio
36	Heff	Effective height (m)
38	*AnalysisNo*	Identifier for a specific analysis case
39	*FrameName*	Name or ID of the structural frameName
40	*EQ_Record_Name*	Name or ID of the earthquake ground-motion record
41	**ADE_frame_disp_max_sofar_m**	Max. frame displacement so far during ADE analysis (m)
42	**ADE_frame_2steps_back_disp_m**	Frame displacement at $t - 2\Delta t$ (m)
43	**ADE_frame_1step_back_disp_m**	Frame displacement at $t - \Delta t$ (m)
44	ADE_frame_disp_m	Targets for Frame displacement (m)

TABLE II: Performance Metrics of All Trained Models

Model	Number of Input Features	Number of Qubits	Bond Dimension, Number of Output Features	Total Trainable Parameters: Classical+Quantum	Dropout	RMSE	MAE	R^2	SMAPE
LSTM-192-3L	36	–	–	769729	0.00	0.0057	0.0026	0.9123	34.2726
LSTM-192-3L	19	–	–	756673	0.00	0.0055	0.0030	0.9205	39.9860
LSTM-192-3L	36	–	–	769729	0.25	0.0052	0.0036	0.9273	45.8032
LSTM-192-3L	19	–	–	756673	0.25	0.0048	0.0032	0.9387	37.9208
LSTM-192-3L	36	–	–	769729	0.30	0.0057	0.0036	0.9144	43.6373
LSTM-192-3L	19	–	–	756673	0.30	0.0072	0.0040	0.8611	48.4953
Bi-LSTM-192-3L	19	–	–	2103169	0.24	0.0028	0.0020	0.9798	26.7314
Bi-LSTM-192-3L	36	–	–	2129281	0.25	0.0069	0.0045	0.8739	49.6516
Bi-LSTM-192-3L	19	–	–	2103169	0.25	0.0031	0.0020	0.9750	26.5607
MPS-LSTM-192-2L	19	–	9,10	468803	0.20	0.0124	0.0101	0.5945	119.4588
MPS-LSTM-192-2L	19	–	9,12	473437	0.20	0.0041	0.0031	0.9544	38.8267
MPS-LSTM-192-2L	19	–	9,14	478071	0.20	0.0052	0.0038	0.9284	47.1893
MPS-LSTM-192-2L	19	–	9,16	482705	0.20	0.0031	0.0022	0.9747	27.7500
MPS-LSTM-192-2L	19	–	6,14	466059	0.20	0.0042	0.0031	0.9537	38.3935
MPS-LSTM-192-2L	19	–	8,14	466059	0.20	0.0027	0.0021	0.9807	27.6690
MPS-LSTM-192-2L	19	–	10,14	483139	0.20	0.0033	0.0024	0.9703	31.2189
MPS-LSTM-192-2L	19	–	12,14	494871	0.20	0.0027	0.0021	0.9806	28.1349
CNN-32-QLSTM-3L-96-Bi-LSTM-96-2L	19	4	–	406169+32	0.10	0.0054	0.0037	0.9238	47.2976
CNN-32-QLSTM-3L-96-Bi-LSTM-96-2L	19	5	–	406404+40	0.10	0.0044	0.0034	0.9487	41.0416
CNN-32-QLSTM-3L-96-Bi-LSTM-96-2L	19	6	–	406641+48	0.10	0.0054	0.0042	0.9237	48.5063
CNN-32-QLSTM-3L-96-Bi-LSTM-96-2L	19	7	–	406880+56	0.10	0.0048	0.0038	0.9401	47.1854
CNN-32-QLSTM-3L-96-Bi-LSTM-96-2L	19	5	–	406404+40	0.00	0.0063	0.0043	0.8933	50.4034
CNN-32-QLSTM-3L-96-Bi-LSTM-96-2L	19	5	–	406404+40	0.20	0.0088	0.0070	0.7941	74.4712
CNN-32-QLSTM-3L-96-LSTM-96-2L-CPLX	19	5	–	171108+120	0.10	0.0058	0.0042	0.9102	52.8189
CNN-32-QLSTM-3L-96-LSTM-96-2L	19	5	–	171108+40	0.10	0.0056	0.0042	0.9182	50.3443
CNN-32-QLSTM-3L-96-LSTM-96-2L-ET	19	5	–	171108+40	0.10	0.0065	0.0049	0.8891	56.4927
CNN-32-QLSTM-3L-96-LSTM-96-2L-M1	19	5	–	171108+80	0.10	0.0066	0.0049	0.8852	55.1445
CNN-32-QLSTM-3L-96-LSTM-96-2L-M2	19	5	–	171108+80	0.10	0.0067	0.0043	0.8820	51.1712
CNN-32-QLSTM-3L-96-LSTM-96-2L-PL	19	5	–	406404+60	0.10	0.0047	0.0033	0.9409	41.2629
CNN-32-QLSTM-3L-96-LSTM-96-2L-PL-ET	19	5	–	406404+60	0.10	0.0037	0.0028	0.9644	33.7526
CNN-32-QLSTM-3L-96-LSTM-96-2L-QRP	19	5	–	406404+60	0.10	0.0045	0.0031	0.9460	42.7493
CNN-32-QLSTM-3L-96-LSTM-96-2L-QRP-ET	19	5	–	406404+60	0.10	0.0054	0.0043	0.9236	54.4934