Nonlinear Dyn (2024) 112:14479–14496
https://doi.org/10.1007/s11071-024-09813-3

RESEARCH

RefreshNet: learning multiscale dynamics through hierarchical refreshing

Junaid Farooq · Danish Rafiq · Pantelis R. Vlachas · Mohammad Abid Bazaz

Received: 4 January 2024 / Accepted: 27 May 2024 / Published online: 10 June 2024

Abstract Forecasting complex system dynamics, particularly for long-term predictions, is persistently hindered by error accumulation and computational burdens. This study presents RefreshNet, a multiscale framework developed to overcome these challenges, delivering an unprecedented balance between computational efficiency and predictive accuracy. RefreshNet incorporates convolutional autoencoders to identify a reduced order latent space capturing essential features of the dynamics, and strategically employs multiple recurrent neural network blocks operating at varying temporal resolutions within the latent space, thus allowing the capture of latent dynamics at multiple temporal scales. The unique "refreshing" mechanism in RefreshNet allows coarser blocks to reset inputs of finer blocks, effectively controlling and alleviating error accumulation. This design demonstrates superiority over existing techniques regarding computational efficiency and predictive accuracy, especially in long-term forecasting. The framework is validated using three benchmark applications: the FitzHugh–Nagumo system, the Reaction–Diffusion equation, and Kuramoto–Sivashinsky dynamics. RefreshNet significantly outperforms state-of-the-art methods in long-term forecasting accuracy and speed, marking a significant advancement in modeling complex systems and opening new avenues in understanding and predicting their behavior.

Keywords Complex systems · Multiscale dynamics · Deep learning · Autoencoder

J. Farooq (✉) · M. A. Bazaz
Department of Electrical Engineering, National Institute of Technology Srinagar, Hazratbal, Srinagar, Jammu and Kashmir 190006, India
e-mail: junaid_phd017@nitsri.ac.in

M. A. Bazaz
e-mail: abid@nitsri.ac.in

D. Rafiq
Department of Electrical Engineering, Islamic University of Science and Technology, Awantipora, Pulwama, Jammu and Kashmir 192122, India
e-mail: danish.rafiq@iust.ac.in

P. R. Vlachas
Institute of Structural Engineering, ETH Zurich, 8093 Zürich, Zürich, Switzerland
e-mail: pvlachas@ethz.ch

1 Introduction

Advanced modeling and simulation methods [1], encompassing data-driven approaches and those grounded on first principles, have significantly influenced a wide array of quantitative scientific fields. This impact is evident in diverse domains including language processing [2], speech modeling [3], financial analytics [4], health informatics [5], biology [6], and epidemiology [7], all of which have reaped substantial benefits from these technologies. These methods are transformative in their capacity to forecast outcomes, particularly in areas where extensive experimentation is impractical or too expensive, often due to high infrastructure, equipment, and skilled personnel requirements. However, a critical challenge remains: the precision of these simulations heavily relies on their ability to comprehensively represent the spatial and temporal scales of the phenomena being studied [8].

Multiscale systems involve many interacting scales, exhibiting nonlinear, interconnected dynamics across a broad scale range. From microscopic to macroscopic levels, the varying granularities at which these systems operate complicate the task of accurate prediction. A striking example of this challenge can be seen in climate modeling [9]. Scientists must account for various dynamic processes, from microscale atmospheric interactions to macroscale global climate phenomena. The system's components are not isolated but interconnected, often resulting in nonlinear and feedback-driven interactions. These features give rise to emergent properties that are not readily predictable from studying individual components in isolation. Therefore, forecasting future spatiotemporal data for long-term horizons remains a formidable challenge in the predictive modeling of complex systems.

Research efforts aimed at addressing the challenges of multiscale modeling and simulation have culminated in the development of the equation-free framework (EFF) [8,10]. EFF facilitates efficient modeling of multiscale high-dimensional systems by integrating coarse-grained and fine-scale simulations [11,12]. However, this approach is not without its own challenges, such as selecting appropriate degrees of freedom for these representations.

Recently, the Learning Effective Dynamics (LED) approach, an advanced extension of the EFF framework that incorporates cutting-edge machine learning architectures into EFF, has shown enhanced predictability over traditional reduced-order models [13]. LED refines the simulation process by operating within a reduced order latent space and then expanding to high-dimensional states through the use of convolutional autoencoders [13]. Originally introduced as a nonlinear alternative to principal component analysis [14], autoencoders (AEs) offer powerful capabilities for dimensionality reduction and feature learning. AEs have been used in the identification of non-linear modes across multiple fields from structural mechanics [15,16], materials modeling [17], fluid flows [18], to general dynamical systems [19]. By capturing nonlinear relationships within the data, AEs can model complex patterns and structures that may not be captured by linear techniques. Notably, LED autoencoders have proven to be effective in establishing mappings between fine and coarse-grained representations, thereby enhancing simulation efficiency. LED has been successfully applied to various systems, from molecular simulations [20] to high-dimensional fluid flows [13].

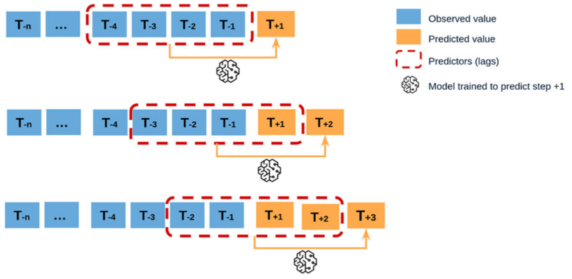


Fig. 1 Demonstration of iterative, autoregressive forecasting

LED employs Recurrent Neural Networks (RNNs) [21,22] to propagate the latent reduced-order dynamics. Such machine learning architectures specialized in sequential data have revolutionized the field of multi-variate time series forecasting, from dynamical systems [23–30] to audio and speech processing [31–35], achieving orders of magnitude better performance compared to traditional methods. Multiple other works, similar to LED, have employed RNNs coupled with autoencoders for spatiotemporal dynamics modeling [16,36–39].

Autoregressive time series algorithms, like RNNs employed in LED, face a critical shortcoming known as the compounding error effect or error accumulation [40–43]. This phenomenon occurs as these models generate future forecasts based on previous predictions, as depicted in Fig. 1. Consequently, errors accumulate at each forecasting step, resulting in diminished forecast quality as we extend further into the future [44]. This so-called error accumulation problem is widely recognized within the research community, and proposed solutions range from novel architectures that alleviate the autoregressive propagation (e.g., Transformers [45]), to regularization techniques [46], and novel training methods [40–43,47–50].

Another factor contributing to error accumulation in autoregressive methods is their training objective, which focuses on minimizing the loss in one-step-ahead predictions. However, during autoregressive testing, these networks base their inferences on selfgenerated predictions, scenarios not encountered in training. This phenomenon, known as the exposure bias effect, can negatively impact the RNN's ability to generalize effectively [51].

While LED has demonstrated promising results in the modeling and simulating of multiscale dynamical systems, it continues to utilize autoregressive RNNs for latent space propagation, inheriting associated limitations such as iterative error propagation and exposure bias effect. The present research is motivated by these persistent challenges in the predictive modeling of complex multiscale systems. In particular, we address these two critical problems that have yet to be solved: the mismatch between the training objective and autoregressive inference and the pervasive issue of error accumulation in long-term forecasting.

In this paper, we propose a novel framework called RefreshNet, specifically designed to address these challenges by employing a hierarchy of RNNs operating at different timescales within the LED framework. RefreshNet is designed to address the issue of accumulating errors, providing greater accuracy than LSTMs and LED while also delivering enhanced computational efficiency over the LED framework. The results of this study promise a leap forward in our ability to forecast the behavior of complex systems over longer horizons, enabling the prediction of the behavior of complex systems across numerous applications. The unique contributions of our framework include the following:

1. Innovative Refreshing Mechanism: Unlike previous models such as [13], our framework introduces a refreshing mechanism that dynamically updates the input to the finer-scale RNN blocks with predictions from coarser-scale blocks. This process significantly mitigates the error accumulation inherent in long-term predictions, which is a notable limitation in existing frameworks including the LED.
2. Hierarchical Multi-Timescale Learning: Our architecture's hierarchical design enables an integrated learning approach across multiple timescales, enhancing the model's ability to understand and predict complex dynamical systems' behavior with unprecedented accuracy and reliability.
3. Error accumulation reduction: RefreshNet incorporates multiple RNN blocks at various temporal resolutions. The coarser blocks refresh the finer blocks' inputs, resetting error accumulation, improving prediction accuracy, and enabling longer prediction times with lower errors.
4. Demonstrated Superior Performance: Through rigorous benchmarking on systems previously explored by LED, our RefreshNet framework demonstrates not incremental, but substantial improvements in forecasting performance. This highlights the practical and theoretical advancements achieved by our methodology.
5. Computational benefits: Our framework has the potential to significantly reduce the computation time by delaying the need to solve equations based on first principles numerically, outperforming state-of-the-art techniques.
6. Multiscale dynamics learning: Integrating multiple temporal resolutions enables effective learning of multiscale dynamics.
7. Scalability: RefreshNet can adapt to complex systems with evolving dynamics by integrating additional RNN blocks at even coarser temporal resolutions.

The remainder of this paper is organized as follows. Section 2 presents the general methodology of the RefreshNet framework. In Sect. 3, we demonstrate the results of RefreshNet applied on benchmark problems of complex dynamical systems forecasting. In Sect. 4, we discuss the implications, limitations, and future scope of the proposed framework. Finally, Sect. 5 concludes the paper.

2 Methods

The high-dimensional state of a dynamical system is given by $\mathbf{x}_t \in \mathbb{R}^{d_x}$, and the discrete time dynamics are given by

$$\mathbf{x}_{t+\Delta t} = \mathcal{F}(\mathbf{x}_t) \tag{1}$$

where Δt is the sampling period and $\mathcal{F}$ may be nonlinear, deterministic or stochastic. The state of the system at time t can also be described by a vector $\mathbf{z}_t \in \mathcal{Z}^{d_z}$, where $\mathcal{Z} \subset \mathbb{R}^{d_z}$ is the low dimension manifold with $d_z << d_x$. To identify this manifold, we define an encoder $\mathcal{E}^{w_\mathcal{E}} : \mathbb{R}^{d_x} \rightarrow \mathbb{R}^{d_x}$, where $w_\mathcal{E}$ represent the trainable parameters. Thus, the high-dimensional state $\mathbf{x}_t$ is transformed to low-dimensional state $\mathbf{z}_t$ such that $\mathbf{z}_t = \mathcal{E}^{w_\mathcal{E}}(\mathbf{x}_t)$. This latent state is mapped back to the

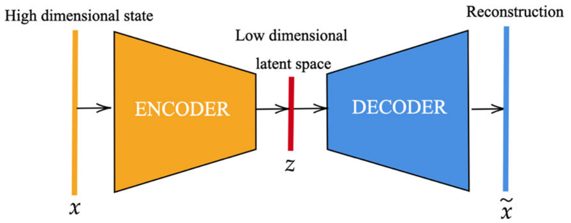


Fig. 2 Classical autoencoder

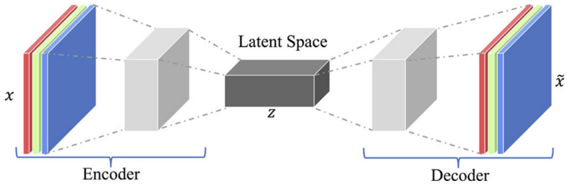


Fig. 3 Convolutional autoencoder

original state using a decoder, i.e., $\tilde{\mathbf{x}}_t = \mathcal{D}^{w_\mathcal{D}}(\mathbf{z}_t)$. The combination of the encoder and the decoder are termed together as autoencoder.

AEs [52] are neural networks that utilize nonlinear transformations to map an input to a lower-dimensional latent space and then reconstruct it back to its original dimension at the output. During training, the objective is to minimize the reconstruction loss, typically measured as the squared difference between the input and the reconstructed output, i.e. $\mathcal{L} = |\mathbf{x} - \tilde{\mathbf{x}}|^2$. This loss function guides the AE to learn meaningful representations of the input data. Figures 2 and 3 provide a visual depiction of an AE, illustrating the flow of information from the input layer to the latent space and back to the output layer.

In this study, we use convolutional neural network (CNN) based AEs. CNNs have been specifically designed to effectively process image data, leveraging the inherent spatial correlations present in such data. The architecture of a CNN comprises multiple layers, each processing a multidimensional input that includes a channel axis and spatial axes. By employing convolutional kernels or filters that slide along the spatial axes of the input, CNNs exploit the spatial structure of the data. This characteristic of CNNs can be seen as a geometric prior, as they inherently consider the structural relationships within the data.

In this study, we incorporate CNN layers within an AE framework, where a bottleneck layer is introduced to reduce dimensionality. The layers commonly employed in a convolutional autoencoder (CAE) include convolutional, pooling, and upsampling layers. Convolutional layers apply filters to capture spatial features in the input data, while pooling layers downsample the spatial dimensions to reduce computational complexity and extract essential information. Upsampling layers, also known as transposed convolutional layers or deconvolutional layers, perform the opposite operation of pooling layers by increasing the spatial dimensions of the data. These layers collectively enable the autoencoder to effectively encode high-dimensional input data into a lower-dimensional latent space and subsequently decode it to reconstruct the original input. The combination of these layers in a convolutional autoencoder allows for the extraction of meaningful features from complex data and facilitates accurate reconstruction of the input data. A typical CAE is represented in Fig. 3.

The optimal parameters of the CAE are determined by minimizing the mean squared reconstruction error (MSE):

$$\begin{aligned} w_\mathcal{E}^*, w_\mathcal{D}^* &= \operatorname{argmin}_{w_\mathcal{E}, w_\mathcal{D}} (\mathbf{x}_t - \tilde{\mathbf{x}}_t)^2 \\ &= \operatorname{argmin}_{w_\mathcal{E}, w_\mathcal{D}} (\mathbf{x}_t - \mathcal{D}^{w_\mathcal{D}}(\mathcal{E}^{w_\mathcal{E}}(\mathbf{x}_t)))^2. \end{aligned} \tag{2}$$

Unlike other dimensionality reduction techniques such as AE, PCA, or Diffusion maps [53] that rely on vectorization of input field data, CNN-based approaches consider the spatial structure of the data. For instance, when an input field is shifted by a pixel, the vectorized version exhibits significant differences, while the convoluted image representation is more robust to such shifts. By incorporating CNN layers in the autoencoder, our proposed approach benefits from the CNN's ability to capture spatial correlations and exploit the geometric prior encoded in the data. This allows for a more comprehensive and accurate representation of the underlying structure in the data, which in turn enhances the effectiveness of the dimensionality reduction process.

To capture non-Markovian effects and preserve memory within the low-order manifold (coarse scale), we employ an RNN, a nonlinear propagator, as the fundamental building block of our framework. The RNN unit learns the forecasting rule as follows:

$$h_t = \mathcal{H}^{w_\mathcal{H}}(\mathbf{z}_t, h_{t-\Delta t}), \tag{3}$$

$$\tilde{\mathbf{z}}_{t+\Delta t} = \mathcal{R}^{w_\mathcal{R}}(h_t) \tag{4}$$

where $h_t \in \mathcal{R}^{d_h}$ is the internal hidden memory state, $\tilde{z}_{t+\Delta t}$ is the latent state prediction, $\mathcal{H}^{w_{\mathcal{H}}}$ and $\mathcal{R}^{w_{\mathcal{R}}}$ are the hidden-to-hidden and hidden-to-output mappings with $w_{\mathcal{H}}$ and $w_{\mathcal{R}}$ as the respective trainable parameters. In this study, we use the Long Short-Term Memory (LSTM) implementation of RNN [22,54], that employs gates to control the information flow and alleviate training problems of previously proposed architectures.

The role of the RNN in our approach is twofold. Firstly, it updates its hidden memory state h_t by considering the current input state $\mathbf{z}_t$ and the previous hidden memory state $h_{t-\Delta t}$. This enables the RNN to effectively track the historical evolution of the low-order state, thereby capturing the non-Markovian dynamics of the system. Secondly, leveraging the updated hidden state h_t, the RNN forecasts the latent state at the next time-step(s), represented as $\tilde{\mathbf{z}}_{t+\Delta t}$. Through training, the RNN is optimized to minimize the forecasting loss, quantified by the squared difference between the predicted latent state $\tilde{\mathbf{z}}_{t+\Delta t}$ and the actual latent state at the corresponding future time-step $\mathbf{z}_{t+\Delta t}$. This optimization process is achieved through BPTT, allowing the RNN to learn and improve its forecasting capabilities over time.

What sets our proposed RefreshNet apart from the LED and other frameworks is its utilization of multiple RNN blocks, hierarchically operating at different timescales. These timescales increase geometrically by a factor of k, thereby capturing varying temporal resolutions. The initial RNN block, denoted as $\mathcal{R}_1$, operates at a fine-grained time scale of one-step (i.e., $\Delta t_1 = 1$), allowing for precise one-step-ahead predictions. It is trained to minimize the one-step ahead prediction loss, using training data with a temporal resolution of 1 unit.

Subsequently, higher-level RNN blocks, such as $\mathcal{R}_2$ and $\mathcal{R}_3$, are incorporated into the hierarchy and trained to predict the system dynamics at coarser temporal resolutions. For instance, $\mathcal{R}_2$ operates with a temporal resolution of $\Delta t_2 = k$, while $\mathcal{R}_3$ operates at an even coarser resolution of $\Delta t_3 = k^2$. The training data is appropriately subsampled to align with these temporal resolutions. This hierarchical arrangement of RNN blocks with increasing temporal resolutions enables the RefreshNet to capture multi-scale dynamics and efficiently model system behavior across different time scales.

In order to understand how this proposed hierarchy of RNNs alleviates the effect of error accumulation, it is crucial to consider that the number of prediction steps undertaken directly influences the accumulation of error within the RNN blocks. Initially, when simulating the system up to the desired point $\mathbf{z}(t)$ with a small number of prediction steps ($s < \epsilon$), the accumulated error remains negligible. However, as the number of prediction steps increases beyond a threshold ($s > \epsilon$), the accumulated error becomes significant after surpassing $s > \epsilon + C$. The specific values of ϵ and C are problem-dependent and rely on various factors, including the system's complexity and the hyperparameters of the RNN, such as the required input sequence length. The input sequence length signifies the number of past values available to the RNN to predict the subsequent state value.

The hierarchical nature of the RNN building blocks described earlier implies that the system simulation up to the desired point $z(t)$ involves varying levels of error accumulation within each RNN block. Specifically, when using $\mathcal{R}_1$ with a time step of $\Delta t_1 = 1$, the number of steps required to reach $z(t)$ is t. On the other hand, when using $\mathcal{R}_2$ with a time step of $\Delta t_2 = k$ and $\mathcal{R}_3$ with a time step of $\Delta t_3 = k^2$, the number of steps to reach the same point is reduced to t/k and t/k^2, respectively.

Assuming a fixed error per timestep and ignoring the increased difficulty of longer-term predictions, the accumulated error in $\mathcal{R}_3$ is significantly lower compared to that in $\mathcal{R}_2$, which, in turn, is notably lower than the accumulated error in $\mathcal{R}_1$. This observation stems from the increasing temporal resolutions and the corresponding ability of each RNN block to capture finer details of the system dynamics. As a result, the hierarchical arrangement of the RNN blocks allows for a more accurate and reliable representation of the system's behavior as we progress through the hierarchy.

Our numerical experiments have revealed an important finding: the value of ϵ is directly influenced by the selection of the input sequence length. This observation can be attributed to the intuitive notion that providing a more extended history to the RNN during training and testing enables it to make predictions further into the future. Consequently, as we increase the input sequence length, the threshold ϵ also increases, indicating that the accumulated error becomes significant after a greater number of prediction steps.

This finding underscores the critical role of the input sequence length in determining the accuracy and reliability of the predictions made by the RNN. By considering a more extended history, the RNN can effectively capture the underlying dynamics and improve its forecasting capabilities, thereby reducing the impact of error accumulation. Therefore, careful consideration and optimization of the input sequence length are crucial for achieving more accurate and longer-term predictions in our framework. In this context, we set the value of k equal to the input sequence length, which remains the same for all the RNN blocks of the hierarchy.

In addition to capturing multiple scales of dynamics, the hierarchical RNN framework incorporates a refreshing mechanism during autoregressive inference that plays a crucial role in minimizing error accumulation. Once the first coarser block, $\mathcal{R}_2$, becomes operational, it is employed to refresh the inputs of the finer block, $\mathcal{R}_1$, effectively eliminating the accumulated error within $\mathcal{R}_1$. The temporal resolution of $\mathcal{R}_2$ is k, and its input sequence length is k. As a result, the minimum simulation time required for $\mathcal{R}_2$ to become operational is $t = k^2$. Similarly, $\mathcal{R}_3$ requires the simulation to have progressed to at least $t = k^3$ to become operational.

Once these coarser blocks become operational, they possess freshness and are virtually error-free as they start their first prediction steps, allowing them to refresh the inputs of $\mathcal{R}_1$. This refreshing process occurs periodically, with $\mathcal{R}_2$ refreshing $\mathcal{R}_1$ every k^2 time steps and $\mathcal{R}_3$ refreshing $\mathcal{R}_1$ every k^3 time steps. This cycle continues as we progress through the hierarchy, ensuring that each block performs optimally and minimizes the effective error accumulation. Figure 4 shows this process's sequential data flow.

Figure 5 offers an exhaustive visual representation of the RefreshNet architecture, detailing the end-to-end flow from the initial condition through to the generation of predictions. This figure represents the operational synergy between the convolutional autoencoder and the hierarchy of RNN blocks, each tuned to a specific temporal scale. The encoder segment of the architecture efficiently compresses the high-dimensional input data into a latent space, which serves as the foundation for the subsequent predictive modeling executed by the RNN blocks. These blocks, structured to cater to different temporal granularities, implement the refreshing mechanism. Through this mechanism, RefreshNet adeptly bridges the gap between fine-grained detail and broad temporal trends, illustrating our novel approach to enhancing the accuracy and reliability of predictions across extended time horizons. The integrated use of a decoder to revert the latent predictions back into their original high-dimensional space further exemplifies the framework's ability to maintain the fidelity of the reconstructed outputs.

By incorporating this refreshing mechanism and considering the appropriate simulation length, the hierarchical RNN framework enables accurate and reliable predictions by mitigating the impact of accumulated errors. Moreover, this mechanism alleviates the exposure bias and the mismatch between training and autoregressive testing and paves the way for forecasting dynamical systems for much longer time scales.

3 Results

In this section, we compare RefreshNet with LED and an LSTM across benchmark prototypical applications [55], including the FitzHugh–Nagumo Model, the Reaction–Diffusion equation, and dynamics derived from the Kuramoto–Sivashinsky equation. All conducted experiments were performed on an Apple M1 Pro, with 16 GB of RAM memory and 8 CPU Cores. Wherever applicable, the parameters and hyperparameters of different autoencoder and RNN blocks used to implement LSTM, LED, and RefreshNet were kept the same as LED [13] to facilitate comparison.

3.1 FitzHugh–Nagumo model (FHN)

RefreshNet is employed to capture the dynamics of the FitzHugh–Nagumo equations (FHN) [56,57]. The FHN model describes the evolution of an activator $u(x, t) = \rho_{\text{ac}}(x, t)$ and an inhibitor density $v(x, t) = \rho_{\text{in}}(x, t)$ on the domain $x \in [0, L]$:

$$\frac{\partial u}{\partial t} = D_u \frac{\partial^2 u}{\partial x^2} + u - u^3 - v,$$
$$\frac{\partial v}{\partial t} = D_v \frac{\partial^2 v}{\partial x^2} + \varepsilon(u - \alpha_1 v - \alpha_0).$$

The system evolves periodically under two timescales, with the activator/inhibitor density acting as the "fast"/"slow" variable, respectively. The bifurcation parameter $\varepsilon = 0.006$ controls the difference in the timescales. The chosen parameter values are: $D_u = 1$, $D_v = 4$, $L = 20$, $\alpha_0 = -0.03$, and $\alpha_1 = 2$, as per [13].

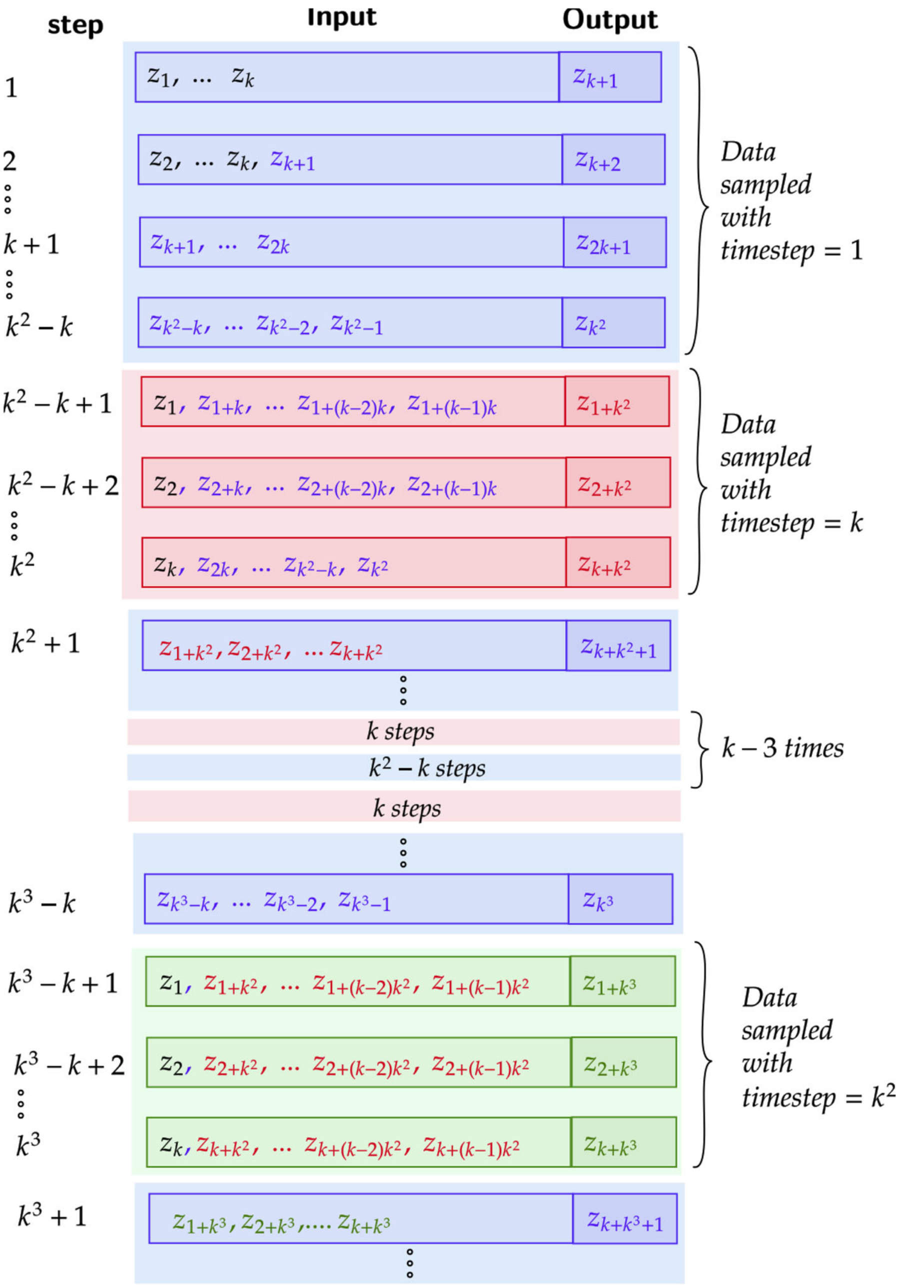

step
Input
Output
1
$z_1, \dots z_k$
z_{k+1}
2
$z_2, \dots z_k, z_{k+1}$
z_{k+2}
$k+1$
$z_{k+1}, \dots z_{2k}$
z_{2k+1}
k^2-k
$z_{k^2-k}, \dots z_{k^2-2}, z_{k^2-1}$
z_{k^2}
Data sampled with timestep = 1
k^2-k+1
$z_1, z_{1+k}, \dots z_{1+(k-2)k}, z_{1+(k-1)k}$
z_{1+k^2}
k^2-k+2
$z_2, z_{2+k}, \dots z_{2+(k-2)k}, z_{2+(k-1)k}$
z_{2+k^2}
k^2
$z_k, z_{2k}, \dots z_{k^2-k}, z_{k^2}$
z_{k+k^2}
Data sampled with timestep = k
k^2+1
$z_{1+k^2}, z_{2+k^2}, \dots z_{k+k^2}$
z_{k+k^2+1}
k steps
k^2-k steps
$k-3$ times
k steps
k^3-k
$z_{k^3-k}, \dots z_{k^3-2}, z_{k^3-1}$
z_{k^3}
k^3-k+1
$z_1, z_{1+k^2}, \dots z_{1+(k-2)k^2}, z_{1+(k-1)k^2}$
z_{1+k^3}
k^3-k+2
$z_2, z_{2+k^2}, \dots z_{2+(k-2)k^2}, z_{2+(k-1)k^2}$
z_{2+k^3}
k^3
$z_k, z_{k+k^2}, \dots z_{k+(k-2)k^2}, z_{k+(k-1)k^2}$
z_{k+k^3}
Data sampled with timestep = k^2
k^3+1
$z_{1+k^3}, z_{2+k^3}, \dots z_{k+k^3}$
z_{k+k^3+1}

Fig. 4 Data sequences at different prediction steps in RefreshNet. Blue, red, and green colors represent the $\mathcal{R}_1$, $\mathcal{R}_2$, and $\mathcal{R}_3$ models and blue, red, and green fonts represents the data generated by them, respectively. Black fond represents the initial input data sequence. For clearer visualization, see Fig. 17 in the Appendix showing a case study for $k = 1$

To discretize the above equations, we utilize a grid with $N = 101$ points and solve them using the Lattice Boltzmann (LB) method [58] with a time-step of $\delta t = 0.005$. For comparison with the results in [13], we employ the LB method to generate data from six different initial conditions, obtaining the fine-grained solution used in this study. The generated data is then subsampled, retaining every 200th data point, resulting in a coarse time step of $\Delta t = 1$. The training set consists of a single time series with 2000 points, the validation set consists of another time series with 2000 points, and the testing set comprises four time series with 10,000 data points each, all originating from different initial conditions. The hyperparameters of the AE and LSTM networks are tuned based on the Mean Squared Error (MSE) calculated on the validation data while main-

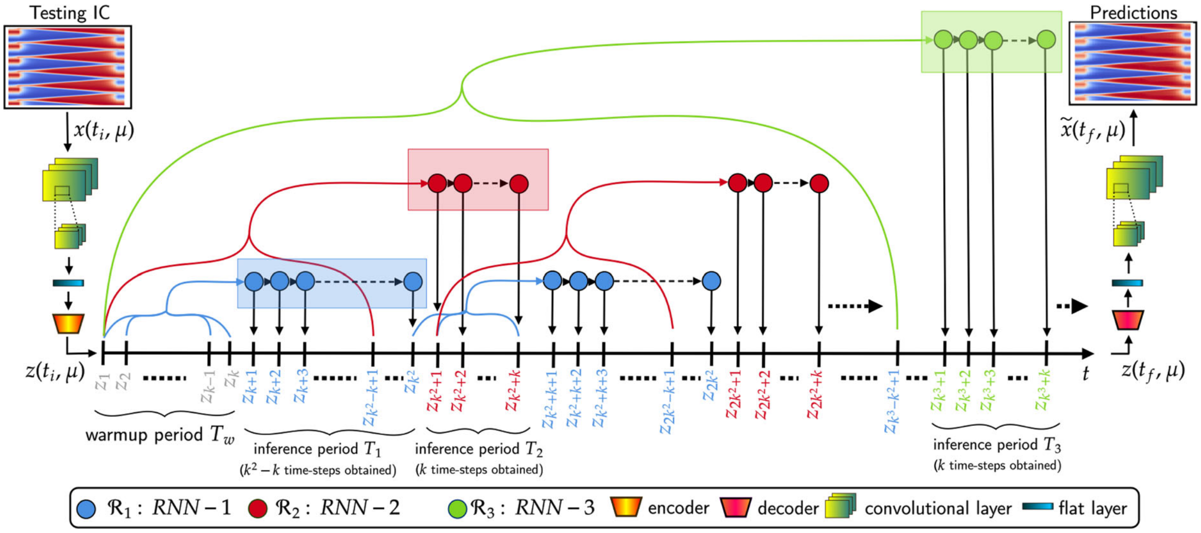

Testing IC
Predictions
$x(t_i, \mu)$
$\tilde{x}(t_f, \mu)$
$z(t_i, \mu)$
$z(t_f, \mu)$
t
z_1 z_2 ...... z_{k-1} z_k
z_{k+1} z_{k+2} z_{k+3} ...... z_{k^2-k+1} ... z_{k^2}
z_{k^2+1} z_{k^2+2} ... z_{k^2+k}
z_{k^2+k+1} z_{k^2+k+2} z_{k^2+k+3} ...... z_{2k^2-k+1} ... z_{2k^2}
z_{2k^2+1} z_{2k^2+2} ... z_{2k^2+k}
...... $z_{k^3-k^2+1}$ ...
z_{k^3+1} z_{k^3+2} z_{k^3+3} ... z_{k^3+k}
warmup period T_w
inference period T_1
($k^2 - k$ time-steps obtained)
inference period T_2
(k time-steps obtained)
inference period T_3
(k time-steps obtained)
● $\mathcal{R}_1 : RNN-1$
● $\mathcal{R}_2 : RNN-2$
● $\mathcal{R}_3 : RNN-3$
encoder
decoder
convolutional layer
flat layer

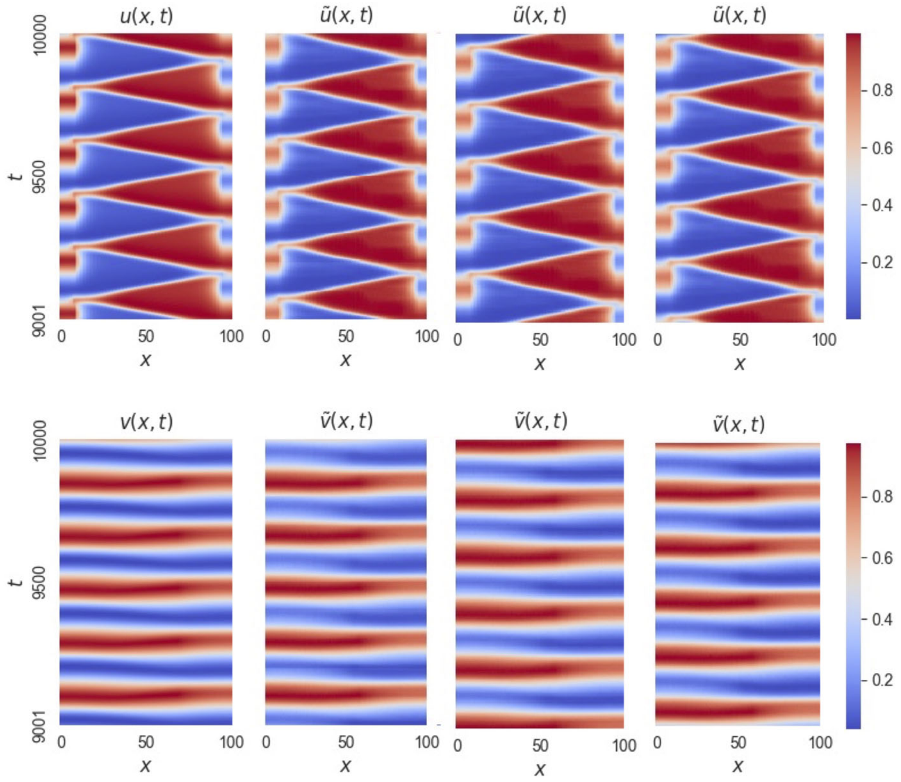
Fig. 5 Comprehensive illustration of the RefreshNet architecture, showcasing the end-to-end process of transforming initial conditions into long-term predictions across multiple temporal scales. The process begins with the encoding of the testing initial condition (IC) into a compact latent vector $z(t_i, \mu)$ via a convolutional autoencoder. This latent representation, capturing the essential dynamics at a reduced dimensionality, is sequentially processed through a hierarchy of RNN blocks, each operating at a distinct temporal resolution. These RNN blocks, ranging from fine-grained to coarse-grained temporal scales, leverage a novel refreshing mechanism to mitigate error accumulation, thereby enhancing the fidelity of long-term forecasts. The latent forecasts are finally decoded by the Decoder block, reconstructing the high-dimensional state predictions. This architecture enables the efficient and accurate modeling of complex dynamical systems by integrating multiscale temporal dynamics and deep learning

taining consistency with [13] for proper comparison. The details of the network are provided in Table 1.

To compare the performance of our proposed method to LED and LSTM, we consider the MSE. Table 6 illustrates the results from $t = 9000$ to $t = 10{,}000$. The results are comprehensively compared in Table 2. The error accumulation is shown in Figs. 7 and 8. Figure 9 shows the evolution of latent dynamics. Notably, even at 10,000 prediction steps, the proposed RefreshNet exhibits strikingly minimal error accumulation. For LED and LSTM, the error accumulates for a certain period and then appears to reduce. This phenomenon is explained by the periodic nature of the states where the difference in the frequencies and phases of the original and predicted trajectories dominates the error. The error reduces once the phase difference crosses half of the time-period mark and increases again after a complete cycle. RefreshNet successfully alleviates the error accumulation problem and demonstrates lower errors than LSTM and the LED.

Table 1 Hyperparameters used to implement LSTM, LED, and RefreshNet for FHN

Specifics	Value
Latent space generator	Autoencoder (AE)
Number of AE layers	{3}
Size of AE layers	{100}
Activation of AE layers	celu
Latent dimension	{2}
AE input/output data scaling	[0,1]
AE output activation	$1 + 0.5tanh(.)$
AE weight decay rate	{0.0}
AE batch size	32
AE initial learning rate	0.001
RNN cell type	lstm
LSTM BPPT sequence length	{10}
Number of RNN layers in each block	{1}
Size of RNN layers	{32}
Activation of RNN cells	tanh(.)
Output activation of RNN cells	$1 + 0.5tanh(.)$



(a) Ground Truth (b) RefreshNet (c) LED (ρ=1) (d) LSTM

Fig. 6 Original trajectories of activator $u(x, t)$ and inhibitor $(v(x, t)$ densities and the corresponding generated trajectories using different methods

Table 2 Accuracy and computational performance compared to the LB method (CPU time: 90.7 s) for FHN model

Method	Computation time (%)	MSE
LED ($\rho = 1$)	51.10	0.2238
LED ($\rho = 10$)	12.40	0.2296
LED ($\rho = 100$)	3.98	0.2201
LSTM	2.78	0.2223
RefreshNet	2.43	0.0041

3.2 Reaction–diffusion equation

In our investigation, we apply RefreshNet to the lambda-omega reaction–diffusion system [59,60] described by the following equations:

$$\frac{\partial u}{\partial t} = [1 - (u^2 + v^2)]u + \beta(u^2 + v^2)v + d_1\nabla^2 u,$$
$$\frac{\partial v}{\partial t} = -\beta(u^2 + v^2)u + [1 - (u^2 + v^2)]v + d_2\nabla^2 v, \tag{5}$$

where $-10 \leq x, y \leq 10$. The reaction parameter is $\beta = 1.0$, and the diffusion parameters are $d_1 = d_2 = 1$. The equations are numerically solved on a 96×96 uniform grid using the fourth-order Runge–Kutta-Fehlberg method with a $\Delta t = 0.05$ time step. The system state is represented by the tensor $w = (u, v) \in \mathbb{R}^{2\times 96\times 96}$. Notably, the system exhibits a spiral wave pattern, with the specific shape influenced by the parameter d. The equation is solved using Euler's method with details mentioned in [55].

The hyperparameters of the CAE and LSTM networks are tuned based on the Mean Squared Error (MSE) calculated on the validation data while maintaining consistency with [13] for proper comparison. Hyper-parameters of the network's architecture are provided in Table 3 (Fig. 11).

To assess the performance of RefreshNet, we employ the mean squared error (MSE) as the metric to measure the error. The simulation is conducted for 10,000 time steps. Figure 10 illustrates the results of our simulations at $t = 10{,}000$ in comparison to the LSTM and the LED. Table 4 presents a comprehensive comparison of the

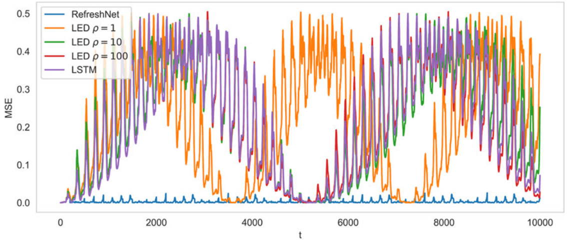


Fig. 7 Error accumulation during FHN prediction in different methods

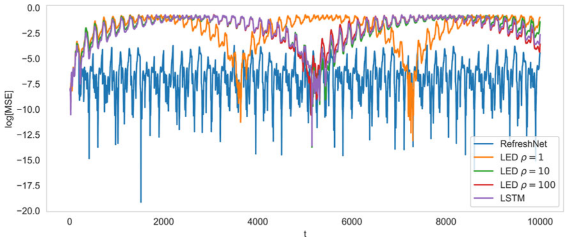


Fig. 8 Error accumulation during FHN prediction in different methods (log scale)

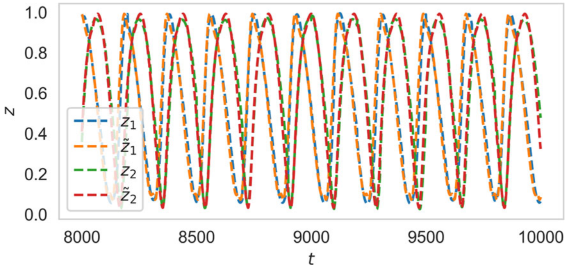


Fig. 9 Evolution of FHN latent dynamics

results. The error accumulation is shown in Figs. 12 and 13. Figure 11 shows the evolution of latent dynamics. Remarkably, the proposed RefreshNet exhibits minimal error accumulation, confirming its effectiveness.

Table 3 Hyperparameters used to implement LSTM, LED, and RefreshNet for RD

Specifics	Value
Latent space generator	2D convolutional autoencoder
Kernels	encoder: 5-5-5-5, Decoder: 5-5-5-5
Channels	2-16-16-16-16-8-16-16-16-16-2
Activation of CNN layers	celu
Latent dimension	{8}
CAE input/output data scaling	[0,1]
CAE output activation	$1 + 0.5tanh(.)$
CAE weight decay rate	{0.0}
CAE batch size	32
CAE initial learning rate	0.001
RNN cell type	lstm
LSTM BPPT sequence length	{10}
Number of RNN layers in each block	{1}
Size of RNN layers	{64}
Activation of RNN cells	tanh(.)
Output activation of RNN cells	$1 + 0.5tanh(.)$

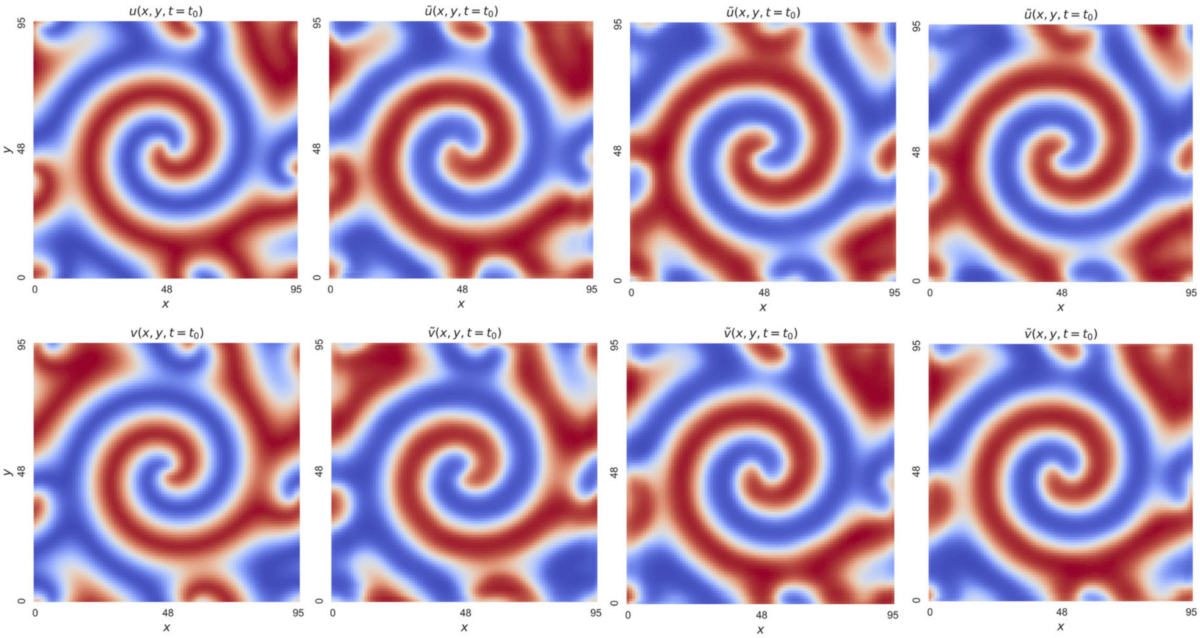
3.3 Kuramoto–Sivashinsky

The Kuramoto–Sivashinsky (KS) [61,62] is a prototypical partial differential equation (PDE) of fourth order that exhibits a very rich range of nonlinear phenomena, and is frequently used as a benachmark of chaotic dynamics [23,24,30,63]. In case of high dissipation and small spatial extent L (domain size), the long-term dynamics of KS can be represented on a low-dimensional inertial manifold [64,65], that attracts all neighboring states at an exponential rate after a transient period. Here, RefreshNet and LED are employed to learn the low-order manifold of the effective dynamics in KS and forecast their long-term evolution.

The one-dimensional K-S equation is given by the PDE:

$$\frac{\partial u}{\partial t} = -\nu \frac{\partial^4 u}{\partial x^4} - \frac{\partial^2 u}{\partial x^2} - u \frac{\partial u}{\partial x}. \tag{6}$$

on the domain $\Omega = [0, L]$ with periodic boundary conditions $u(0, t) = u(L, t)$ and $\nu = 1$. The special case $L = 22$ considered in this work is studied extensively in [66] and exhibits a structurally stable chaotic attractor, i.e., an inertial manifold where the long-term dynamics lie. Equation (6) is discretized with a grid of size 64 points and solved using the fourth-order method for stiff PDEs introduced in [67] with a time-step of $\delta t = 2.5 \times 10^{-3}$ starting from a random initial condition. The data is subsampled to $\Delta t = 0.25$ (coarse time-step of RefreshNet and LED). 15×10^3 samples are used for training, and another 15×10^3 for validation. For testing purposes, the process is repeated with a different random seed, generating another 15×10^3 samples.



(a) Ground Truth (b) RefreshNet (c) LED (ρ=1) (d) LSTM

Fig. 10 Original trajectories of RD equation and the corresponding generated trajectories using different methods at $t = 10{,}000$

Table 4 Accuracy and computational performance of different methods for RD model compared to the original solution [55] (CPU time: 203.0 s))

Method	Computation time (%)	MSE
LED ($\rho = 1$)	52.1	0.07
LSTM	1.8	0.2692
RefreshNet	1.9	0.0054

The hyperparameters of the CAE and LSTM networks (components of the RefreshNet and LED frameworks) are tuned based on the Mean Squared Error (MSE) calculated on the validation data while maintaining consistency with [13] for proper comparison. The details of the network are provided in Table 5.

To compare the performance of our proposed method with LED, we consider the MSE. Figures 14, 15, and 16 qualitatively illustrate the results up to $t = 50$, consistent with [13]. Notably, the maximum Lyapunov exponent of the KS equation for spatial extend $L = 22$ is approximately 0.043 [68], implying a Lyapunov time of 23.25 unit times. We observe that prediction after this time deteriorates, which is expected due to the chaoticity of the system, as even minimal prediction errors close to machine precision accumulate exponentially. Table 6 provides a comprehensive comparison of the results. RefreshNet demonstrates a 41% reduction in Mean Squared Error (MSE) compared to LSTM, with a similar computational cost, highlighting its significant efficiency gains. Compared to LED, RefreshNet achieves a 30% improvement in MSE and operates an order of magnitude faster.

4 Discussion

The hierarchical multi-timescale RNN framework proposed in this study presents a unique and powerful approach to capturing the dynamics of complex systems with multiple time scales. By incorporating multiple RNN blocks at different temporal resolutions, the framework captures fine-grained details and takes advantage of coarse-grained information to refresh and refine predictions.

This refreshing mechanism is a crucial feature that sets our framework apart from traditional RNN models.

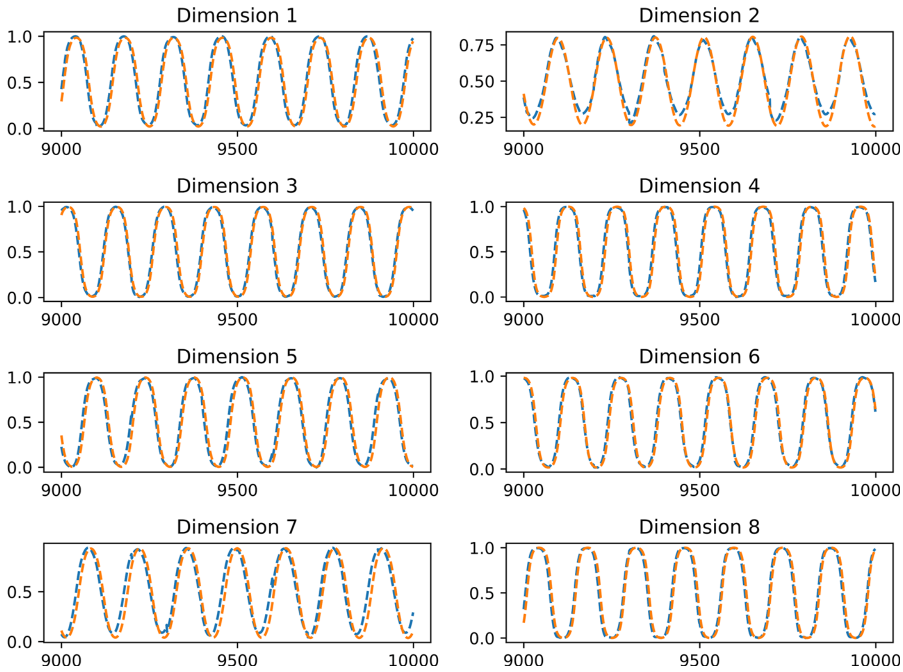


Fig. 11 Evolution of latent dynamics in RD model using RefreshNet. Each subplot shows the original and the predicted values

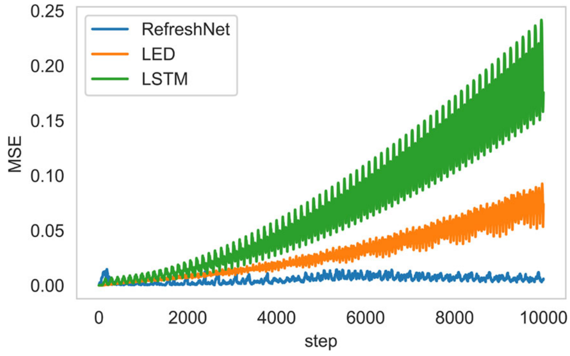


Fig. 12 Error accumulation during RD prediction in different methods

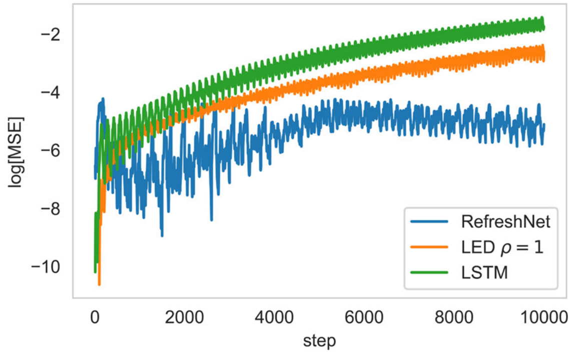


Fig. 13 Error accumulation during RD prediction in different methods (log scale)

Table 5 Hyperparameters used to implement LSTM, LED, and RefreshNet for KS

Specifics	Value
Latent space generator	1D convolutional autoencoder
Kernels	encoder: 5-5-5-5, Decoder: 5-5-5-5
Channels	1-16-32-64-8-8-8-64-32-16-1
Activation of CNN layers	celu
Latent dimension	{8}
CAE input/output data scaling	[0,1]
CAE output activation	$1 + 0.5tanh(.)$
CAE weight decay rate	{0.0}
CAE batch size	32
CAE initial learning rate	0.001
RNN cell type	lstm
LSTM BPPT sequence length	{10}
Number of RNN layers in each block	{1}
Size of RNN layers	{512}
Activation of RNN cells	tanh(.)
Output activation of RNN cells	$1 + 0.5tanh(.)$

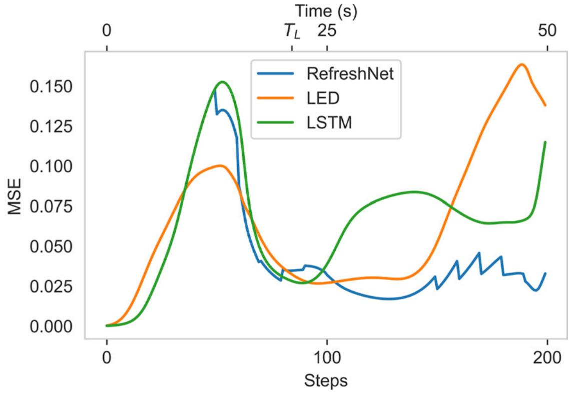


Fig. 14 Error accumulation during KS prediction using different methods ($T_L = 20.83$ is the Lyapunov time)

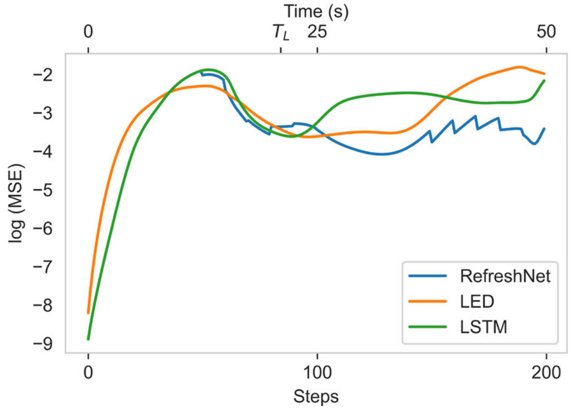


Fig. 15 Error accumulation (log scale) during KS prediction using different methods ($T_L = 20.83$ is the Lyapunov time)

The coarser blocks' ability to refresh the finer blocks' inputs effectively resets the error accumulation, leading to more accurate and reliable predictions. This process allows the framework to achieve a delicate balance between capturing intricate details and minimizing the impact of accumulated errors, resulting in superior performance compared to conventional RNN models.

Furthermore, the hierarchical nature of the framework provides a natural hierarchy of scales, with each RNN block operating at a different temporal resolution. This hierarchical structure enables the framework to adaptively capture the dynamics at various scales, ranging from fine-scale fluctuations to broader trends. By incorporating multiple scales in the modeling process, our framework offers a more comprehensive understanding of complex systems and their evolution.

In contrast to the state-of-the-art LED method, the proposed framework alleviates error accumulation and enables much larger prediction times. Due to the error accumulation, the LED framework would resort to simulating the original high-dimensional dynamics early on, increasing its computational cost and time to solution. In contrast, the proposed framework leverages the capability of a hierarchy of multi-timescale RNN blocks of learning the dynamics and refreshing the prediction in autoregressive inference, alleviating the need for explicit numerical simulations and thus achieving significant reductions in computational time. This reduction in computation time is highly advantageous, especially when dealing with large-scale complex systems or performing real-time predictions.

The benefits of the hierarchical RNN framework extend beyond accurate prediction. The framework's modular and scalable nature allows for easy integration of additional RNN blocks at even coarser temporal resolutions, enabling dynamics modeling across a broader range of scales. This flexibility and scalability make our framework well-suited for studying complex systems with evolving dynamics, such as climate systems, financial markets, and biological processes.

By combining accuracy on long-term forecasting and scalability, our proposed framework surpasses LED and other existing methods in capturing the long-term dynamics of complex multiscale systems. It offers an innovative and efficient approach that can unlock new possibilities for studying and understanding complex phenomena across various domains.

By leveraging convolutional autoencoders and recurrent neural networks, our framework captures the intricate dynamics of complex systems and learns a compressed latent space representation by LED. This latent space, encoding the essential features and patterns of the complex system, holds the potential for generative modeling applications, expanding our framework's potential beyond traditional forecasting tasks. It enables us to explore creative possibilities, such as scenario generation, data augmentation, and synthetic data generation for training purposes. The ability to generate new samples that capture the essence of the complex system's dynamics contributes to a more comprehensive understanding of the system and facilitates hypothesis testing and scenario analysis. RefreshNet establishes a unique fusion of data-driven methods and first-principles models, unlocking the potential for precise and resource-efficient prediction of complex mul-

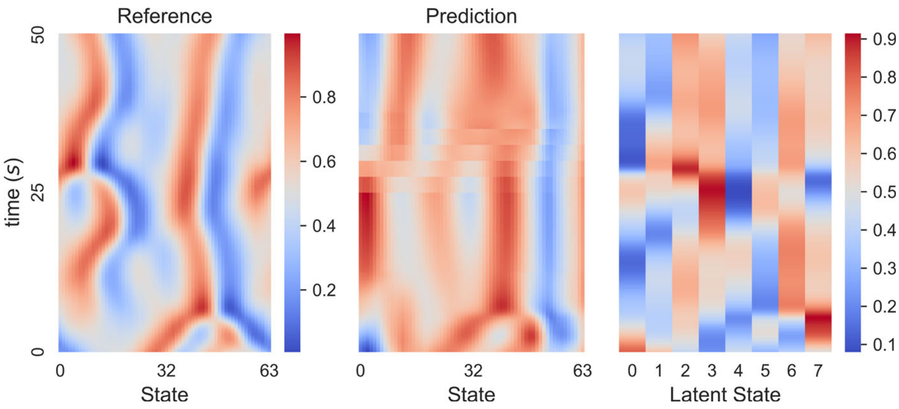


Fig. 16 Evolution of reference, predicted, latent dynamics in KS model using RefreshNet

Table 6 Accuracy and computational performance of different methods for KS model compared to the numerical simulation [67] (CPU time: 61.0 s)

Method	Computation time (%)	MSE
LED ($\rho = 0.25$)	83.1	0.07410
LSTM	2.1	0.08916
RefreshNet	2.3	0.0522

tiscale systems. This framework can be applied to problems in which data is generated from first principles or collected from sensors.

The theoretical rationale of RefreshNet lies in its unique refreshing mechanism and the hierarchical integration of RNN blocks operating across multiple temporal scales. Traditional RNNs, including their variants like LSTMs, face significant challenges in capturing long-term dependencies within time series data, particularly for complex systems exhibiting multiscale temporal dynamics. The vanishing gradient problem, although mitigated to some extent by LSTMs, still poses limitations on the model's ability to learn and remember information over extended sequences. RefreshNet's multi-timescale learning approach theoretically extends the memory capacity of RNNs by structuring prediction tasks across different temporal resolutions. This hierarchical design allows the model to retain and leverage information over longer periods, enhancing its ability to predict long-term system behavior accurately. A critical issue in long-term forecasting with autoregressive models is the accumulation of prediction errors over time. Each prediction step inherently carries a risk of error, and when predictions are sequentially dependent on previous forecasts, these errors can compound, leading to significant deviations from true trajectories. The refreshing mechanism introduced by RefreshNet addresses this issue by periodically updating the inputs to finer-scale RNN blocks with more accurate predictions from coarser-scale blocks. This process effectively resets the accumulation of errors, theoretically improving the fidelity of long-term forecasts. By integrating broader temporal dynamics into finer-scale predictions, RefreshNet mitigates the impact of error accumulation, a novel contribution beyond the capabilities of traditional RNN architectures.

The theoretical foundation of RefreshNet also includes the premise that complex dynamical systems exhibit behaviors and patterns that can be more effectively modeled when considered across multiple temporal scales. Traditional single-scale approaches may overlook or inadequately capture these dynamics, leading to suboptimal forecasting performance. RefreshNet's architecture is designed to capture and leverage these multiscale temporal dynamics, theoretically enabling a more comprehensive and nuanced understanding of the system's behavior.

While our proposed framework offers significant advancements and numerous benefits, there are certain limitations to consider and areas for future research. Firstly, the reliance on the latent space obtained through convolutional autoencoders assumes that it adequately represents the dynamics of the complex system. Exploring alternative generative modeling approaches, such as variational autoencoders or generative adversarial networks, can enhance representation capabilities. Additionally, extending the framework to consider external influences or perturbations would enhance its applicability to a broader range of systems. Incorporating external inputs and developing methods to integrate exogenous factors can improve the framework's robustness and adaptability. Handling data limitations, such as missing or noisy data, can be addressed through data imputation or noise reduction approaches while incorporating domain knowledge and expert guidance can enhance the framework's performance. RefreshNet is designed as a method that operates across multiple timescales. Its capabilities can be enhanced through integration with multiscale autoencoders [69], enabling it to effectively tackle problems that have multiscale characteristics in the spatial domain.

5 Conclusion

In this paper, we have presented a data-driven deep learning model called RefreshNet that effectively captures the dynamics of complex systems across multiple temporal resolutions. By incorporating convolutional autoencoders and multiple recurrent neural network (RNN) blocks operating at geometrically increasing timescales, our framework leverages the power of fine-grained details and coarse-grained information to improve long-term predictions of complex system dynamics.

We have demonstrated that the refreshing mechanism in our framework, where coarser blocks refresh the inputs of finer blocks, plays a crucial role in resetting error accumulation and improving long-term prediction accuracy. This feature sets our framework apart from traditional RNN models and the LED framework and contributes to its superior performance in capturing complex system dynamics.

Through experiments and comparisons with state-of-the-art techniques such as LED on three applications, namely the FitzHugh Nagumo system, the Reaction–Diffusion equation, and the Kuramoto–Sivashinsky dynamics, we have shown that our framework effectively alleviateds the issue of error accumulation, leading to more reliable and accurate predictions. These results demonstrate the potential of the RefreshNet framework to advance the field of complex system modeling.

The hierarchical nature of our framework enables the adaptive modeling of dynamics at different scales, capturing both fine-scale fluctuations and broader trends. This flexibility is further enhanced by our framework's modularity and scalability, allowing for the easy integration of additional RNN blocks at even coarser temporal resolutions. As a result, our framework is well-suited for studying complex systems with evolving dynamics. Moreover, in contrast to LED, alleviating the need to numerically solve the original equations, which entail an exhaustive computational burden, drastically reduces the computation time for our framework.

The proposed RefreshNet framework opens up exciting possibilities for further research and applications in various domains. Its ability to capture complex system behavior across multiple scales offers a more comprehensive understanding of these systems and their evolution. This framework has the potential to contribute to advancements in fields such as climate science, financial markets, and biological processes, where the accurate modeling of complex dynamics is crucial.

In conclusion, the hierarchical refresh framework for multiscale learning presented in this paper represents a significant advancement in capturing the dynamics of complex systems. By leveraging convolutional autoencoders, multiple RNN blocks, and the refreshing mechanism, our framework offers improved prediction accuracy, eliminates error accumulation, and enables the modeling of dynamics at various resolutions. We believe this framework will inspire further research in the field and find valuable applications in understanding and predicting the behavior of complex systems. The results for the KS model suggest RefreshNet's utility in modeling chaotic systems, a potential avenue we plan to explore in the future.

Funding First author acknowledges the funding from the Ministry of Education, Government of India. Second author acknowledges the funding from the Science and Engineering Research Board (SERB), Department of Science and Technology (DST), Government of India via Grant No. PDF/2022/002081.

Data availability The datasets generated during and/or analysed during the current study are not publicly available but are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare no Conflict of interest.

Appendix

Here, data sequences for RefreshNet are shown for a case of $k = 10$ in Fig. 17.

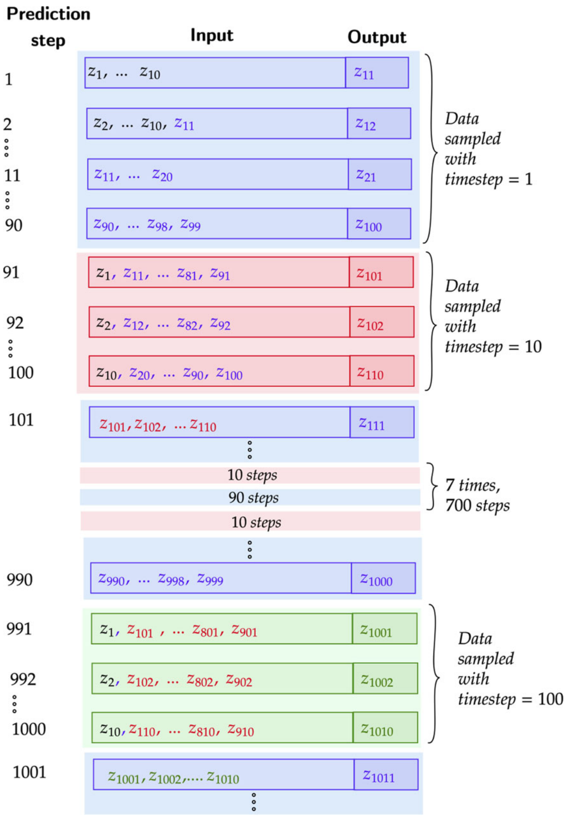

Prediction step
Input
Output
1
$z_1, \dots z_{10}$
z_{11}
2
$z_2, \dots z_{10}, z_{11}$
z_{12}
11
$z_{11}, \dots z_{20}$
z_{21}
90
$z_{90}, \dots z_{98}, z_{99}$
z_{100}
Data sampled with timestep = 1
91
$z_1, z_{11}, \dots z_{81}, z_{91}$
z_{101}
92
$z_2, z_{12}, \dots z_{82}, z_{92}$
z_{102}
100
$z_{10}, z_{20}, \dots z_{90}, z_{100}$
z_{110}
Data sampled with timestep = 10
101
$z_{101}, z_{102}, \dots z_{110}$
z_{111}
10 steps
90 steps
10 steps
7 times, 700 steps
990
$z_{990}, \dots z_{998}, z_{999}$
z_{1000}
991
$z_1, z_{101}, \dots z_{801}, z_{901}$
z_{1001}
992
$z_2, z_{102}, \dots z_{802}, z_{902}$
z_{1002}
1000
$z_{10}, z_{110}, \dots z_{810}, z_{910}$
z_{1010}
Data sampled with timestep = 100
1001
$z_{1001}, z_{1002}, \dots z_{1010}$
z_{1011}

Fig. 17 Data sequences at different prediction steps in RefreshNet for the case $k = 1$. Blue, red, and green colors represent the $\mathcal{R}_1$, $\mathcal{R}_2$, and $\mathcal{R}_3$ models and blue, red, and green fonts represents the data generated by them, respectively. Black fond represents the initial input data sequence

References

1. Zeigler, B.P., Praehofer, H., Kim, T.G.: Theory of Modeling and Simulation. Academic Press, Cambridge (2000)
2. Jozefowicz, R., Vinyals, O., Schuster, M., Shazeer, N., Wu, Y.: Exploring the limits of language modeling. arXiv preprint arXiv:1602.02410 (2016)
3. Deng, L., Yu, D., Acero, A.: Structured speech modeling. IEEE Trans. Audio Speech Lang. Process. **14**(5), 1492–1504 (2006)
4. Benninga, S.: Financial Modeling. MIT Press, Cambridge (2014)
5. Ravì, D., Wong, C., Deligianni, F., Berthelot, M., Andreu-Perez, J., Lo, B., Yang, G.-Z.: Deep learning for health informatics. IEEE J. Biomed. Health Inform. **21**(1), 4–21 (2016)
6. Alber, M., Buganza Tepole, A., Cannon, W.R., De, S., Dura-Bernal, S., Garikipati, K., Karniadakis, G., Lytton, W.W., Perdikaris, P., Petzold, L., et al.: Integrating machine learning and multiscale modeling-perspectives, challenges, and opportunities in the biological, biomedical, and behavioral sciences. NPJ Digit. Med. **2**(1), 115 (2019)
7. Frauenthal, J.C.: Mathematical Modeling in Epidemiology. Springer, Cham (2012)
8. Kevrekidis, I.G., Gear, C.W., Hummer, G.: Equation-free: the computer-aided analysis of complex multiscale systems. AIChE J. **50**(7), 1346–1355 (2004)
9. Dunlea, E., Elfring, C.: A National Strategy for Advancing Climate Modeling. Technical report, National Research Council (2012)
10. Kevrekidis, I.G., Gear, C.W., Hyman, J.M., Kevrekidis, P.G., Runborg, O., Theodoropoulos, C., et al.: Equation-free, coarse-grained multiscale computation: enabling microscopic simulators to perform system-level analysis. Commun. Math. Sci. **1**(4), 715–762 (2003)
11. Laing, C.R., Frewen, T., Kevrekidis, I.G.: Reduced models for binocular rivalry. J. Comput. Neurosci. **28**, 459–476 (2010)
12. Maulik, R., Lusch, B., Balaprakash, P.: Reduced-order modeling of advection-dominated systems with recurrent neural networks and convolutional autoencoders. Phys. Fluids **33**(3), 037106 (2021)
13. Vlachas, P.R., Arampatzis, G., Uhler, C., Koumoutsakos, P.: Multiscale simulations of complex systems by learning their effective dynamics. Nat. Mach. Intell. **4**(4), 359–366 (2022)
14. Moore, B.: Principal component analysis in linear systems: controllability, observability, and model reduction. IEEE Trans. Autom. Control **26**(1), 17–32 (1981)
15. Simpson, T., Dervilis, N., Chatzi, E.: On the use of nonlinear normal modes for nonlinear reduced order modelling. arXiv preprint arXiv:2007.00466 (2020)
16. Simpson, T., Dervilis, N., Chatzi, E.: Machine learning approach to model order reduction of nonlinear systems via autoencoder and LSTM networks. J. Eng. Mech. **147**(10), 04021061 (2021)
17. He, X., He, Q., Chen, J.-S.: Deep autoencoders for physics-constrained data-driven nonlinear materials modeling. Comput. Methods Appl. Mech. Eng. **385**, 114034 (2021)
18. Milano, M., Koumoutsakos, P.: Neural network modeling for near wall turbulent flow. J. Comput. Phys. **182**(1), 1–26 (2002)
19. Otto, S.E., Rowley, C.W.: Linearly recurrent autoencoder networks for learning dynamics. SIAM J. Appl. Dyn. Syst. **18**(1), 558–593 (2019)
20. Vlachas, P.R., Zavadlav, J., Praprotnik, M., Koumoutsakos, P.: Accelerated simulations of molecular systems through learning of effective dynamics. J. Chem. Theory Comput. **18**(1), 538–549 (2021)
21. Elman, J.L.: Finding structure in time. Cogn. Sci. **14**(2), 179–211 (1990)
22. Greff, K., Srivastava, R.K., Koutník, J., Steunebrink, B.R., Schmidhuber, J.: LSTM: a search space odyssey. IEEE Trans. Neural Netw. Learn. Syst. **28**(10), 2222–2232 (2016)

23. Pathak, J., Hunt, B., Girvan, M., Lu, Z., Ott, E.: Model-free prediction of large spatiotemporally chaotic systems from data: a reservoir computing approach. Phys. Rev. Lett. **120**(2), 024102 (2018)
24. Vlachas, P.R., Byeon, W., Wan, Z.Y., Sapsis, T.P., Koumoutsakos, P.: Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. Proc. R. Soc. A Math. Phys. Eng. Sci. **474**(2213), 20170844 (2018)
25. Chang, B., Chen, M., Haber, E., Chi, E.H.: AntisymmetricRNN: a dynamical system view on recurrent neural networks. arXiv preprint arXiv:1902.09689 (2019)
26. Trischler, A.P., D'Eleuterio, G.M.: Synthesis of recurrent neural networks for dynamical system simulation. Neural Netw. **80**, 67–78 (2016)
27. Wan, Z.Y., Vlachas, P., Koumoutsakos, P., Sapsis, T.: Data-assisted reduced-order modeling of extreme events in complex dynamical systems. PLoS One **13**(5), 0197704 (2018)
28. Kundu, A., Ghosh, S., Chakraborty, S.: A long short-term memory based deep learning algorithm for seismic response uncertainty quantification. Probab. Eng. Mech. **67**, 103189 (2022)
29. Zhang, R., Chen, Z., Chen, S., Zheng, J., Büyüköztürk, O., Sun, H.: Deep long short-term memory networks for nonlinear structural seismic response prediction. Comput. Struct. **220**, 55–68 (2019)
30. Vlachas, P.R., Pathak, J., Hunt, B.R., Sapsis, T.P., Girvan, M., Ott, E., Koumoutsakos, P.: Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics. Neural Netw. **126**, 191–217 (2020)
31. Salinas, D., Flunkert, V., Gasthaus, J., Januschowski, T.: DeepAR: probabilistic forecasting with autoregressive recurrent networks. Int. J. Forecast. **36**(3), 1181–1191 (2020)
32. Oord, A.v.d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., Kavukcuoglu, K.: Wavenet: a generative model for raw audio. arXiv preprint arXiv:1609.03499 (2016)
33. Hizlisoy, S., Yildirim, S., Tufekci, Z.: Music emotion recognition using convolutional long short term memory deep neural networks. Eng. Sci. Technol. Int. J. **24**(3), 760–767 (2021)
34. Sak, H., Senior, A.W., Beaufays, F.: Long short-term memory recurrent neural network architectures for large scale acoustic modeling (2014)
35. Graves, A., Mohamed, A.-R., Hinton, G.: Speech recognition with deep recurrent neural networks. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 6645–6649. IEEE (2013)
36. Zang, H., Liu, L., Sun, L., Cheng, L., Wei, Z., Sun, G.: Short-term global horizontal irradiance forecasting based on a hybrid CNN-LSTM model with spatiotemporal correlations. Renew. Energy **160**, 26–41 (2020)
37. O'Donncha, F., Hu, Y., Palmes, P., Burke, M., Filgueira, R., Grant, J.: A spatio-temporal LSTM model to forecast across multiple temporal and spatial scales. Eco. Inform. **69**, 101687 (2022)
38. Ding, Y., Zhu, Y., Feng, J., Zhang, P., Cheng, Z.: Interpretable spatio-temporal attention LSTM model for flood forecasting. Neurocomputing **403**, 348–359 (2020)
39. Hasegawa, K., Fukami, K., Murata, T., Fukagata, K.: CNN-LSTM based reduced order modeling of two-dimensional unsteady flows around a circular cylinder at different Reynolds numbers. Fluid Dyn. Res. **52**(6), 065501 (2020)
40. Sangiorgio, M., Dercole, F.: Robustness of LSTM neural networks for multi-step forecasting of chaotic time series. Chaos Solitons Fractals **139**, 110045 (2020)
41. Teutsch, P., Mäder, P.: Flipped classroom: effective teaching for time series forecasting. arXiv preprint arXiv:2210.08959 (2022)
42. Vlachas, P.R., Koumoutsakos, P.: Learning from predictions: fusing training and autoregressive inference for long-term spatiotemporal forecasts. arXiv preprint arXiv:2302.11101 (2023)
43. Bengio, Y., Louradour, J., Collobert, R., Weston, J.: Curriculum learning. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 41–48 (2009)
44. Selim, M., Zhou, R., Feng, W., Alam, O.: Reducing error propagation for long term energy forecasting using multivariate prediction. In: CATA, pp. 161–169 (2020)
45. Lin, T., Wang, Y., Liu, X., Qiu, X.: A survey of transformers. AI Open (2022)
46. Alzubaidi, L., Zhang, J., Humaidi, A.J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M.A., Al-Amidie, M., Farhan, L.: Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. J. Big Data **8**, 1–74 (2021)
47. Brenner, M., Hess, F., Mikhaeil, J.M., Bereska, L.F., Monfared, Z., Kuo, P.-C., Durstewitz, D.: Tractable dendritic RNNs for reconstructing nonlinear dynamical systems. In: International Conference on Machine Learning, pp. 2292–2320. PMLR (2022)
48. Hess, F., Monfared, Z., Brenner, M., Durstewitz, D.: Generalized teacher forcing for learning chaotic dynamics. arXiv preprint arXiv:2306.04406 (2023)
49. Brenner, M., Koppe, G., Durstewitz, D.: Multimodal teacher forcing for reconstructing nonlinear dynamical systems. arXiv preprint arXiv:2212.07892 (2022)
50. Durstewitz, D., Koppe, G., Thurm, M.I.: Reconstructing computational system dynamics from neural data with recurrent neural networks. Nat. Rev. Neurosci., 1–18 (2023)
51. Peussa, A., Damskägg, E.-P., Sherson, T., Mimilakis, S.I., Juvela, L., Gotsopoulos, A., Välimäki, V.: Exposure bias and state matching in recurrent neural network virtual analog models. In: 2021 24th International Conference on Digital Audio Effects (DAFx), pp. 284–291. IEEE (2021)
52. Baldi, P.: Autoencoders, unsupervised learning, and deep architectures. In: Proceedings of ICML Workshop on Unsupervised and Transfer Learning, pp. 37–49. JMLR Workshop and Conference Proceedings (2012)
53. Nadler, B., Lafon, S., Coifman, R.R., Kevrekidis, I.G.: Diffusion maps, spectral clustering and reaction coordinates of dynamical systems. Appl. Comput. Harmon. Anal. **21**(1), 113–127 (2006)
54. Sherstinsky, A.: Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. Physica D **404**, 132306 (2020)
55. Rafiq, D., Bazaz, M.A.: A collection of large-scale benchmark models for nonlinear model order reduction. Arch. Comput. Methods Eng. **30**(1), 69–83 (2023)

56. Fitzhugh, R.: Computation of impulse initiation and saltatory conduction in a myelinated nerve fiber. Biophys. J . **2**(1), 11–21 (1962)
57. Nagumo, J., Arimoto, S., Yoshizawa, S.: An active pulse transmission line simulating nerve axon. Proc. IRE **50**(10), 2061–2070 (1962)
58. Karlin, I.V., Ansumali, S., Frouzakis, C.E., Chikatamarla, S.S.: Elements of the lattice Boltzmann method I: linear advection equation. Commun. Comput. Phys. **1**(4), 616–655 (2006)
59. Champion, K., Lusch, B., Kutz, J.N., Brunton, S.L.: Data-driven discovery of coordinates and governing equations. Proc. Natl. Acad. Sci. **116**(45), 22445–22451 (2019)
60. Floryan, D., Graham, M.D.: Data-driven discovery of intrinsic dynamics. Nat. Mach. Intell. **4**(12), 1113–1120 (2022)
61. Kuramoto, Y.: Diffusion-induced chaos in reaction systems. Prog. Theor. Phys. Suppl. **64**, 346–367 (1978)
62. Ashinsky, G.S.: Nonlinear analysis of hydrodynamic instability in laminar flames—I. Derivation of basic equations. In: Dynamics of Curved Fronts, pp. 459–488. Elsevier, Amsterdam (1988)
63. Pathak, J., Lu, Z., Hunt, B.R., Girvan, M., Ott, E.: Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data. Chaos Interdiscip. J. Nonlinear Sci. **27**(12), 121102 (2017)
64. Robinson, J.C.: Inertial manifolds for the Kuramoto–Sivashinsky equation. Phys. Lett. A **184**(2), 190–193 (1994)
65. Linot, A.J., Graham, M.D.: Deep learning to discover and predict dynamics on an inertial manifold. Phys. Rev. E **101**(6), 062209 (2020)
66. Cvitanović, P., Davidchack, R.L., Siminos, E.: On the state space geometry of the Kuramoto–Sivashinsky flow in a periodic domain. SIAM J. Appl. Dyn. Syst. **9**(1), 1–33 (2010)
67. Kassam, A.-K., Trefethen, L.N.: Fourth-order time-stepping for stiff PDEs. SIAM J. Sci. Comput. **26**(4), 1214–1233 (2005)
68. Edson, R.A., Bunder, J.E., Mattner, T.W., Roberts, A.J.: Lyapunov exponents of the Kuramoto–Sivashinsky PDE. ANZIAM J. **61**(3), 270–285 (2019)
69. Liu, Y., Ponce, C., Brunton, S.L., Kutz, J.N.: Multiresolution convolutional autoencoders. J. Comput. Phys. **474**, 111801 (2023)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.