entropy

MDPI

Article

Entanglement-Structured LSTM Boosts Chaotic Time Series Forecasting

Xiangyi Meng [1,2] **and Tong Yang** [3,*]

[1] Center for Complex Network Research and Department of Physics, Northeastern University, Boston, MA 02115, USA; x.meng@neu.edu or xm@bu.edu
[2] Department of Physics, Boston University, Boston, MA 02215, USA
[3] Department of Physics, Boston College, Chestnut Hill, MA 02467, USA
* Correspondence: yangto@bc.edu

Abstract: Traditional machine-learning methods are inefficient in capturing chaos in nonlinear dynamical systems, especially when the time difference Δt between consecutive steps is so large that the extracted time series looks apparently random. Here, we introduce a new long-short-term-memory (LSTM)-based recurrent architecture by tensorizing the cell-state-to-state propagation therein, maintaining the long-term memory feature of LSTM, while simultaneously enhancing the learning of short-term nonlinear complexity. We stress that the global minima of training can be most efficiently reached by our tensor structure where all nonlinear terms, up to some polynomial order, are treated explicitly and weighted equally. The efficiency and generality of our architecture are systematically investigated and tested through theoretical analysis and experimental examinations. In our design, we have explicitly used two different many-body entanglement structures—matrix product states (MPS) and the multiscale entanglement renormalization ansatz (MERA)—as physics-inspired tensor decomposition techniques, from which we find that MERA generally performs better than MPS, hence conjecturing that the learnability of chaos is determined not only by the number of free parameters but also the tensor complexity—recognized as how entanglement entropy scales with varying matricization of the tensor.

Keywords: quantum entanglement; recurrent neural networks; tensorization; chaotic dynamical system; chaotic time series forecasting

Citation: Meng, X.; Yang, T. Entanglement-Structured LSTM Boosts Chaotic Time Series Forecasting. *Entropy* **2021**, *23*, 1491. https://doi.org/10.3390/e23111491

Academic Editor: Rosario Lo Franco

Received: 4 October 2021
Accepted: 6 November 2021
Published: 11 November 2021

1. Introduction

Time series forecasting [1], despite its undoubtedly tremendous potential in both theoretical issues (e.g., mechanical analysis, ergodicity) and real-world applications [2] (e.g., traffic, weather, and clinical records analysis), has long been known as an intricate field. From classical work on statistics such as auto-regressive moving average (ARMA) families [3] and basic hidden Markov models (HMM) [4,5] to contemporary machine-learning (ML) methods [6–9] such as gradient boosted trees (GBT) and neural networks (NN), the essential complexity in time series has been more and more frequently recognized. In particular, forecasting models have extended their applicable range from linear, Markovian cases to nonlinear, non-Markovian, and even more general situations [10]. Among all known methods, recurrent NN architectures [11], including plain recurrent neural networks (RNN) [12] and long short-term memory (LSTM) [13], are the most capable of capturing this complexity, as they admit the fundamental recurrent behavior of time series data. LSTM has proved useful in speech recognition and video analysis tasks [14] in which maintaining long-term memory is essential to the complexity. In relation to this objective, novel architectures such as higher-order RNN/LSTM (HO-RNN/LSTM) [15] have been introduced to capture long-term non-Markovianity explicitly, further improving performance and leading to more accurate theoretical analysis.

Still, another domain of complexity—chaos—has been far less understood [16,17]. Even though enormous theory/data-driven studies on forecasting chaotic time series by means of recurrent NN have been conducted [18–25], there is still a lack of consensus on which features play the most important roles in the forecasting methods. The notorious indication of chaos,

$$|\delta x_t| \approx e^{\lambda t}|\delta x_0| \tag{1}$$

(where λ denotes the spectrum of Lyapunov exponents), suggests that the difficulty of forecasting chaotic time series is two-fold: first, any small error will propagate exponentially, and thus multi-step-ahead predictions will be exponentially worse than one-step-ahead ones; second, and more subtly, when the actual time difference Δt between consecutive steps increases, the minimum redundancy of model capacity needed for smoothly descending to the global minima (or sufficiently good local ones) during NN training also increases exponentially. Most studies only address the first difficulty by improving the prediction accuracy achievable at the global minima. Yet the latter is in fact more crucial, especially when Δt is so large that the time series looks apparently random and a trivial local minimum would most likely be reached instead. Recently, tensorization has been introduced in recurrent NN architectures [26,27]. A tensorized version of HO-RNN/LSTM, namely, HOT-RNN/LSTM [28], has claimed an advantage in learning long-term nonlinearity in Lorenz systems of small Δt. On the one hand, we believe that the global minima of chaos (where the dominance of linear dependence is absent) can be most efficiently reached through tensorization approaches, where all nonlinear terms, up to some polynomial order, are treated explicitly and weighted equally. On the other hand, for simple chaotic dynamical systems, nonlinear complexity is only encoded in the short term, not the long term, which HO/HOT models will not be efficient in capturing when Δt is large. Hence, a new tensorization-based recurrent NN architecture is desired so as to foster our understanding of chaos in time series and to meet practical needs, e.g., the modeling of laminar flame fronts and chemical oscillations [29–32].

In this paper, we introduce a new LSTM-based architecture by tensorizing the cell-state-to-state propagation therein, retaining the long-term memory features of LSTM while simultaneously enhancing the learning of short-term nonlinear complexity. Compared with traditional LSTM architectures, including stacked LSTM [33] and other aforementioned statistics/ML-based forecasting methods, our model is shown to be a general and outperforming approach for capturing chaos in almost every typical chaotic continuous-time dynamical system and discrete-time map with controlled comparable NN training conditions, justified by both our theoretical analysis and experimental results. Our model is also tested on real-world time series datasets, where the improvements range up to 6.3%.

During the tensorization, we have explicitly embedded many-body quantum state structures—a way of reducing the exponentially large degree of freedom of a tensor (i.e., tensor decomposition)—popularly studied in condensed matter physics, which is not unseen in NN design [34]. A many-body entangled state living in a tensor-product Hilbert space is hardly separable. The same inseparability behavior also appears in nonlinear multivariate functions when crossing terms between different variables become too complex. This similarity motivated us to adopt a special measure of tensor complexity, namely, the entanglement entropy (EE) [35], which is commonly used in quantum physics and quantum information [34]. For one-dimensional many-body states, two thoroughly studied, popular but different structures exist—multiscale entanglement renormalization ansatz (MERA) [36] and matrix product states (MPS) [37], of which the EE scales with the subsystem size or not at all, respectively [35]. For most pertinent studies, MPS has been proven to be efficient enough to be applicable to a variety of tasks [38–41]. However, our experiments show that, regarding our entanglement-structured design of the new tensorized LSTM architecture, LSTM-MERA performs even better than LSTM-MPS in general without increasing the number of parameters. This finding leads to another interesting result. We conjecture

that not only should tensorization be introduced, but the tensor's EE has to scale with the system size as well; hence, MERA is more efficient than MPS in learning chaos.

2. Recurrent Architecture and Tensorization

2.1. Formalism of LSTM Architecture

The formalism starts from an operator-theoretical perspective by defining two general types of real *operators*, $\mathcal{W}$ and σ, through which most NN architectures can be represented. $\mathcal{W} : \mathbb{X} \to \mathbb{G}$ is simply a linear operator, but $\sigma : \mathbb{G} \to \mathbb{G}$ is a nonlinear operator that $\sigma(G) = (\sigma \circ G) \in \mathbb{G}$ given $G \in \mathbb{G}$ where $\circ$ stands for the entry-wise operator product. All double-struck symbols $(\mathbb{X}, \mathbb{G}, \cdots)$ used in this context are general real vector spaces considered to be of *covariant* type, as $\mathcal{W}$ can be interpreted as a linear-map-induced 2-*contravariant* bilinear form. Next, a *state propagation function* (i.e,. a gate) $g(x, y, \cdots; \mathcal{W}) = \sigma(\mathcal{W}(x \oplus y \oplus \cdots))$ is introduced, where $x \oplus y \oplus \cdots$ stands for the tensor direct sum of real vectors $x, y, \cdots$. Following the formalism, an LSTM architecture can be expressed as follows:

$$\begin{aligned} s_t &= g(1, x_{t-1}, s_{t-1}; \mathcal{W}_o) \circ \sigma(c_t), \qquad x_t = g(1, s_t; \mathcal{W}_x), \\ c_t &= g(1, x_{t-1}, s_{t-1}; \mathcal{W}_f) \circ c_{t-1} \\ &+ g(1, x_{t-1}, s_{t-1}; \mathcal{W}_i) \circ g(1, x_{t-1}, s_{t-1}; \mathcal{W}_m), \end{aligned} \tag{2}$$

where the four gates controlled by $\mathcal{W}_i$, $\mathcal{W}_m$, $\mathcal{W}_f$, and $\mathcal{W}_o$ are the input, memory, forget, and output gates. The state s_t and the cell state c_t are h-dimensional *covectors*, whereas the input x_t is a d-dimensional covector (Figure 1). Therefore, $\mathcal{W}_i$ (as well as $\mathcal{W}_m$, $\mathcal{W}_f$, and $\mathcal{W}_o$) has a direct-sum contravariant realization as a matrix $W_i \in \mathbf{M}(h, 1) \oplus \mathbf{M}(h, d) \oplus \mathbf{M}(h, h)$ that contains $h(1 + d + h)$ free real parameters at most. During NN training, only these free parameters of the linear operators are learnable, whereas all σ (i.e., activation functions) are fixed to be tanh, sigmoid, or other nonlinear functions. The cell state c_t is designed to suffer less from the vanishing gradient problem and thus to capture long-term memory better, whereas s_t tends to capture short-term dependence.

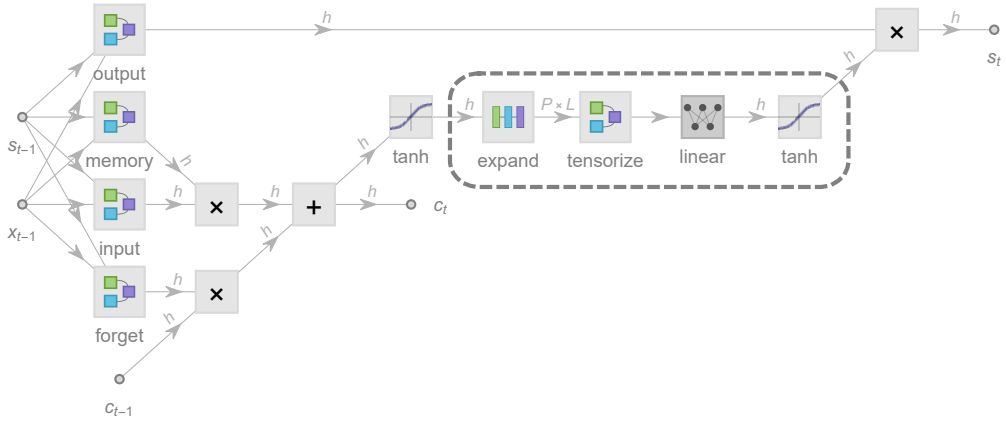


Figure 1. Architecture of a long short-term memory (LSTM) unit in the most common form of four gates: input (i), memory (m), forget (f), and output (o), enhanced by tensorized state propagation with four additional layers embedded (dashed rectangle): *expand* [Equation (4)], *tensorize* (Figure 2), *linear*, and a *tanh* activation function. d is the input dimension of x_t, and h is the hidden dimension of state s_t and cell state c_t. An h-dimensional vector $\tanh c_t$ is first expanded into a $P \times L$-dimensional matrix where L and P are dubbed the physical length and physical degrees of freedom (DOF), respectively. Then, the matrix is tensorized into an L-rank tensor of dimension P^L and passed forward. The effectiveness of this architecture is investigated in Section 4.3.

2.2. Tensorized State Propagation

Our tensorized LSTM architecture (Figure 1) is exactly based on Equation (2), from which the only change is:

$$s_t = g(1, x_{t-1}, s_{t-1}; \mathcal{W}_o) \circ g(\mathcal{T}(\sigma(c_t)); \mathcal{W}_\mathcal{T}). \tag{3}$$

$g(\mathcal{T}(\sigma(c_t)); \mathcal{W}_\mathcal{T})$ is coined a *tensorized state propagation function*, for which $\mathcal{W}_\mathcal{T} : \mathbb{T} \to \mathbb{G}$ acts on as a covariant tensor

$$\mathcal{T}(\sigma(c_t)) = \bigotimes_l (1 \oplus q_{t,l}) = \bigotimes_l (1 \oplus \mathcal{W}_l(\sigma(c_t))). \tag{4}$$

Each $\mathcal{W}_l$ in Equation (4) maps from $\sigma(c_t)$ to a new covector $q_{t,l} \in \mathbb{Q}$. Here, $\mathbb{Q}$ is named a *local q-space*, as, by way of analogy, considered as encoding the *local* degree of freedom in quantum mechanics. $\mathbb{Q}$ can be extended to the complex number field if necessary. Mathematically, Equation (4) offers the possibility of directly constructing orthogonal polynomials up to order L from $\sigma(c_t)$ to build up nonlinear complexity. In fact, when L goes to infinity, $\mathbb{T} = \lim_{L\to\infty} (1 \oplus \mathbb{Q})^{\otimes L} = 1 \oplus \mathbb{Q} \oplus \mathbb{Q} \otimes \mathbb{Q} \oplus \cdots$ becomes a tensor algebra (up to a multiplicative coefficient), and $\mathcal{T}(\sigma(c_t))$ admits any nonlinear smooth function of $\sigma(c_t)$.

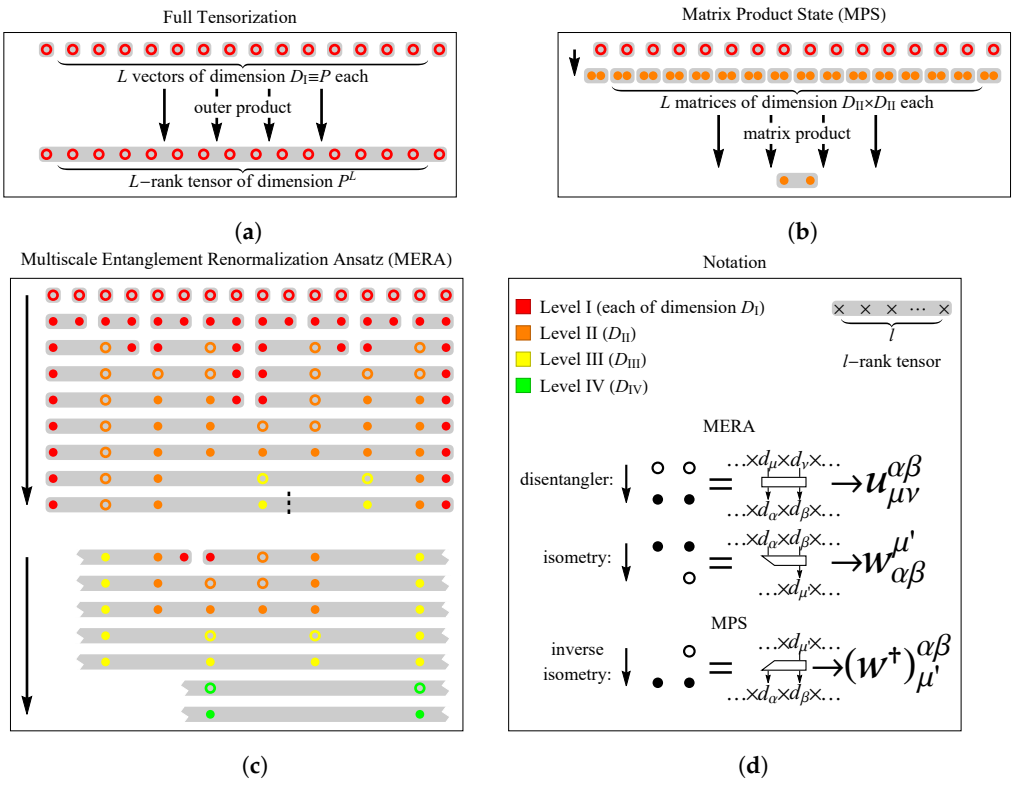


Figure 2. Tensorize layer: quantum entanglement structures. (**a**) Full tensorization. (**b**) Matrix product state (MPS). (**c**) Multiscale entanglement renormalization ansatz (MERA). The MPS and MERA are tensor representations that are widely used for characterizing many-body quantum entanglement in condensed matter physics. (**d**) Notations. A full tensor can be represented by introducing multiple auxiliary and learnable tensors (e.g., disentanglers and isometries, as used in MERA, and inverse isometries, as used in MPS) of different virtual dimensions $\{D_\mathrm{I}, D_\mathrm{II}, \cdots\}$ labeled by different levels, rendered in different colors. The first-level virtual dimension is $D_\mathrm{I} \equiv P$, the physical DOF by definition. Other virtual dimensions $\{D_\mathrm{II}, \cdots\}$ are free hyperparameters to be chosen, the larger of which should better represent the full tensor. The numbers of applicable levels in (**a**,**b**) are always constant (one and two, respectively), yet the number of applicable levels in c is $\log_2 L$, depending on the physical length L.

We now realize Equation (4) by choosing L independent realizations, $W_l \in \mathbf{M}(P-1, h)$, $l = 1, 2, \cdots, L$, which in total contain $L(P-1)h$ learnable parameters at most, each mapping $\sigma(c_t) \equiv \tanh c_t$ to a $(P-1)$-dimensional covector $q_{t,l}$,

$$\begin{pmatrix} [\tanh c_t]_1 \\ [\tanh c_t]_2 \\ \vdots \\ [\tanh c_t]_h \end{pmatrix} \to \begin{pmatrix} \begin{pmatrix} 1 \\ [q_t]_{21} \\ [q_t]_{31} \\ \vdots \\ [q_t]_{P1} \end{pmatrix} & \begin{pmatrix} 1 \\ [q_t]_{22} \\ [q_t]_{32} \\ \vdots \\ [q_t]_{P2} \end{pmatrix} & \cdots & \begin{pmatrix} 1 \\ [q_t]_{2L} \\ [q_t]_{3L} \\ \vdots \\ [q_t]_{PL} \end{pmatrix} \end{pmatrix}. \tag{5}$$

Following Equation (4), $\mathcal{T}(\tanh c_t)$ is simply the tensor product of all column vectors on the right hand side of Equation (5).

From the realization of Equation (3), $W_{\mathcal{T}} \in \mathbf{M}(h, P^L)$ [Figure 2a], however, a problem of exponential explosion (a.k.a. the "curse of dimensionality") arises. Treating $W_{\mathcal{T}}$ maximally by training all hP^L learnable parameters is very computationally expensive, especially as L cannot be small because it governs the nonlinear complexity. To overcome this "curse of dimensionality", *tensor decomposition* techniques have to be exploited [39] for the purpose of finding a much smaller subset $\mathbf{T} \subset \mathbf{M}(h, P^L)$ to which all possible $W_{\mathcal{T}}$ belong, without sacrificing any expressive power.

2.3. Many-Body Entanglement Structures

Below, we introduce the two many-body quantum state structures (MPS and MERA) as efficient low-order tensor decomposition techniques for representing $W_{\mathcal{T}}$.

2.3.1. MPS

As one of the most commonly used tensor decomposition techniques, MPS is also widely known as tensor-train decomposition [42] and takes the following form [Figure 2b]

$$[W_{\mathcal{T}}]^h_{\mu_1 \cdots \mu_L} = \sum_{\{\alpha\}}^{D_{\mathrm{II}}} [w_0]^h_{\alpha_1 \alpha_{L+1}} \left([w_1^\dagger]^{\alpha_1 \alpha_2}_{\mu_1} [w_2^\dagger]^{\alpha_2 \alpha_3}_{\mu_2} \cdots [w_L^\dagger]^{\alpha_L \alpha_{L+1}}_{\mu_L} \right)$$

in our model, where $w_1^\dagger, w_2^\dagger, \cdots, w_L^\dagger$ are learnable 3-tensors (the symbol † denoting that they are *inverse isometries* [35]). D_{II} is an artificial dimension (the same for all α). w_0 is no more than a linear transformation that collects the boundary terms and maintains symmetry. The above notations are used for consistency with quantum theory [35] and the following MERA representation.

2.3.2. MERA

The best way to explain MERA is using graphical tools, e.g., tensor networks [35]. MERA differs from MPS in its hierarchical tree structure: within each level $\{\mathrm{I}, \mathrm{II}, \cdots\}$, the structure contains a layer of 4-tensor *disentanglers* of dimensions $\{D_{\mathrm{I}}^4, D_{\mathrm{II}}^4, \cdots\}$, and then a layer of 3-tensor *isometries* of dimensions $\{D_{\mathrm{I}}^2 \times D_{\mathrm{II}}, D_{\mathrm{II}}^2 \times D_{\mathrm{III}}, \cdots\}$, of which details can be found in [36]. MERA is similar to the Tucker decomposition [43] but fundamentally different because of the existence of disentanglers, which smear the inhomogeneity of different tensor entries [36].

Figure 2c shows the reorganized version of MERA used in our model, where the storage of independent tensors is maximally compressed before they are multiplied with each other by tensor products, which allows more GPU acceleration during NN training.

2.3.3. Scaling Behavior of EE

Now we take advantage of an important measure of tensor complexity: the *entanglement entropy* (EE). Given an arbitrary tensor $W_{\mu_1\cdots\mu_L}$ of dimension P^L and a cut l so that $1 \leq l \ll L$, the EE is defined in terms of the α-Rényi entropy [35],

$$S_\alpha(l) \equiv S_\alpha(W(l)) = \frac{1}{1-\alpha} \log \frac{\sum_{i=1}^{P^l} \sigma_i^\alpha(W(l))}{\left(\sum_{i=1}^{P^l} \sigma_i(W(l))\right)^\alpha}, \tag{6}$$

assuming $\alpha \geq 1$. The Shannon entropy is recovered under $\alpha \to 1$. $\sigma_i(W(l))$ in Equation (6) is the i-th singular value of matrix $W(l) = W_{(\mu_1\times\cdots\times\mu_l),(\mu_{l+1}\times\cdots\times\mu_L)}$, matricized from $W_{\mu_1\cdots\mu_L}$. How $S(l)$ scales with l determines how much redundancy exists in $W_{\mu_1\cdots\mu_L}$, which in turn reveals how efficient at most a tensor decomposition technique can be. For one-dimensional gapped low-energy quantum states (ground states), their EE is saturated even as l increases, i.e., $S_\alpha(l) = \Theta(1)$. Thus, their low-entanglement characteristics can be efficiently represented via MPS, of which the EE does not scale with l either and is bounded by $S_\alpha(l) \leq S_1(l) \leq 2\log D_{\text{II}}$ [35]. By contrast, a non-trivial scaling behavior $S_\alpha(l) = \Theta(\log l)$ corresponds to gapless low-energy states and can only be efficiently represented by MERA, of which $S_\alpha(l) \leq S_1(l) \leq C + \sum_{\text{level}=1}^{\log_2 l} \log D_{\text{level}} \approx C + C' \log l$ scales logarithmically [36]. The bounds of both MPS and MERA have also been proven to be tight [35,36].

The different EE scaling behaviors of MERA and MPS have hence provided an apparent geometric advantage of MERA, i.e., its quasi-two-dimensional structure [Figure 2c], enlarging which will increase not only the width but also the *depth* of NN as the number of applicable levels scales logarithmically with L, offering even more power for model generalization on the already-inherited LSTM architecture [11]. Such an advantage is further confirmed by Equation (9) and then in Section 4.1, in which tensorized LSTMs with the two different representations LSTM-MPS and LSTM-MERA are tested.

3. Theoretical Analysis

3.1. Expressive Power

First, we prove the following theorem that links the variations of c_t and s_t:

Theorem 1. *Given an LSTM architecture [Equation (2)], to which the input is a chaotic dynamical system x_t, characterized by a matrix λ of which the spectrum is the Lyapunov exponent(s), so that any variation δx_t propagates exponentially [Equation (1)], then, up to the first order (i.e., δx_{t-1}),*

$$|\delta s_t| \geq Ce^\lambda |\delta c_t|, \tag{7}$$

where $C \propto 1/\|\mathcal{W}\|_\infty^2$ and $\|\cdot\|_{p=\infty}$ is the operator norm.

Proof. From Equation (2), one has $\delta x_t = (\partial g(1, s_t; \mathcal{W}_x)/\partial s_t)\delta s_t$ where the first-order derivative is bounded by $\|\partial g(1, s_t; \mathcal{W}_x)/\partial s_t\|_\infty \leq \|\mathcal{W}_x\|_\infty \|\sigma'\|_{L_\mu^\infty} \leq \|\mathcal{W}_x\|_\infty$, since the derivative of the active function supported on $(-\infty, \infty)$ satisfies $\|\sigma'\|_{L_\mu^\infty} \equiv \|1/\cosh^2\|_{L_\mu^\infty} \leq 1$. On the other hand, one has

$$\begin{aligned}\delta c_t = \Big[&c_{t-1} \circ \partial g(1, x_{t-1}, s_{t-1}; \mathcal{W}_f)/\partial x_{t-1} + \partial(g(1, x_{t-1}, s_{t-1}; \mathcal{W}_i) \\ &\circ g(1, x_{t-1}, s_{t-1}; \mathcal{W}_m))/\partial x_{t-1}]\delta x_{t-1} + O(\delta x_{t-2}) + \cdots,\end{aligned}$$

and thus

$$|\delta c_t| \leq \|\mathcal{W}_f\|_\infty |c_{t-1}| \circ \delta|x_{t-1}| + (\|\mathcal{W}_i\|_\infty + \|\mathcal{W}_m\|_\infty)\delta|x_{t-1}|,$$

which yields Equation (7), where w.l.o.g. all linear maps are assumed to be around the same magnitude $\sim \|\mathcal{W}\|_\infty$. Note that $|c_{t-1}|$ is also bounded because $|g(1, x_{t-1}, s_{t-1}; \mathcal{W}_f)| \leq 1$, which means c_t is stationary. □

Equation (7) suggests that the state propagation from c_t to s_t carries the chaotic behavior. In fact, to preserve the long-term memorization in LSTM, c_t has to depend on c_{t-1} with a linear behavior and thus cannot carry chaos itself. This is further experimentally verified in Section 4.3. Nevertheless, note that Equation (1) is a necessary condition of chaos, not a sufficient condition.

Now, we look into the expressive power of our introduced tensorized state propagation function, $\mathcal{W}_\mathcal{T}\mathcal{T}(\sigma(c_t))$. One of the advantages of tensorizing the state propagation function in the form of Equation (4) is the well-behaved polynomial space constructed by the tensor product, by virtue of which the approximation of $\mathcal{W}_\mathcal{T}\mathcal{T}$ to any (k-Sobolev) function f can always be bounded, as proven by the following theorem:

Theorem 2. *Let $f \in H^k_\mu(\Lambda)$ be a target function living in the k-Sobolev space, $H^k_\mu(\Lambda) = \left\{ f \in L^2_\mu(\Lambda) \middle| \sum_{|\mathbf{i}|\leq k} \|\partial^{(\mathbf{i})} f\|_{L^2_\mu(\Lambda)} < \infty \right\}$, where $\partial^{(\mathbf{i})} f \in L^2_\mu(\Lambda)$ is the $\mathbf{i}$-th weak derivative of f, up to order $k \geq 0$, square-integrable on support $\Lambda = (-1,1)^h$ with measure μ. $\mathcal{W}_\mathcal{T}\mathcal{T}(\sigma(c_t))$ can approximate $f(\sigma(c_t))$ with an $L^2_\mu(\Lambda)$ error at most*

$$\|f - \mathcal{W}_\mathcal{T}\mathcal{T}\|_{L^2_\mu(\Lambda)} \leq C \min(L, \lfloor L(P-1)/h \rfloor)^{-k} \|f\|_{H^k_\mu(\Lambda)} \tag{8}$$

provided that $(h-1)hP^L \geq (h^{1+\min(L,\lfloor L(P-1)/h \rfloor)} - 1)$. $\|f\|_{H^k_\mu(\Lambda)} = \sum_{|\mathbf{i}|\leq k} \|\partial^{(\mathbf{i})} f\|_{L^2_\mu(\Lambda)}$ is the Sobolev norm and C is a finite constant.

Proof. The *Hölder-continuous spectral convergence theorem* [44] states that $\|f - \mathrm{P}_N f\|_{L^2_\mu(\Lambda)} \leq CN^{-k} \|f\|_{H^k_\mu(\Lambda)}$, in which $\mathrm{P}_N : L^2_\mu(\Lambda) \to \mathbb{P}_N$ is an orthogonal projection that maps f to $\mathrm{P}_N f$. $\sigma(c(t)) \in \Lambda$ is guaranteed as $\sigma \equiv \tanh$. The Sobolev space $\mathbb{P}_N \subset L^2_\mu(\Lambda)$ is spanned by polynomials of a degree of at most N. Next, note that in the realization of $\mathcal{T}(\sigma(c_t))$ each W_l is independent [Equation (4)], and thus $\mathbb{P}_N = \mathrm{span}(\mathcal{T}(\sigma(c_t)))$ is possible, where N is determined by L, P, and h. When $P - 1 \geq h$, the maximum polynomial order is guaranteed L; when $P - 1 < h$, $\dim\{\mathbb{Q}\} < \dim\{\mathbb{G}\}$, and hence $\mathcal{T}(\sigma(c_t))$ can only fully cover a polynomial order of up to $\lfloor L(P-1)/h \rfloor$. Finally, Equation (8) is proven based on the fact that $\mathcal{W}_\mathcal{T}\mathcal{T}$ maximally admits $\mathrm{P}_N f$ as long as $hP^L \geq \sum_{i=0}^N h^i = (h^{N+1} - 1)/(h-1)$, the latter of which is the size of the maximum orthogonal polynomial basis admitted by $\mathbb{P}_N$. □

Equation (8) can be used to estimate how L scales with the chaos the dynamical system possesses. In particular, Equation (7) suggests that $\partial^{(1)} f \sim e^{\lambda \Delta t}$, where Δt is the actual time difference between consecutive steps. Therefore, to persevere the error bound [Equation (8)] one at least expects that $L^{-1} \sim e^{-\lambda \Delta t}$, i.e., L has to increase exponentially with respect to $\lambda \Delta t$. To achieve this, tensorization is undoubtedly the most efficient approach, especially when Δt is large.

3.2. Worst-Case Bound by EE

The above analysis offers an intuitive bound on the expressive power of $\mathcal{W}_\mathcal{T}\mathcal{T}$. Unfortunately, Equation (8) is valid only when all hP^L degrees of freedom of $\mathcal{W}_\mathcal{T}$ are independent. A low-order tensor decomposition may therefore impact the expressive power.

Below, we compare the two different entanglement structures, MPS and MERA, which have major differences in their EE scaling behaviors. We proceed via the following theorem, relating the tensor approximation error to entanglement scaling:

Theorem 3. *Given a tensor* $[W_{\mathcal{T}}]_{\mu_1 \cdots \mu_L}$ *and its tensor decomposition* $\overline{W}_{\mathcal{T}}$*, the worst-case p-norm* ($p \geq 1$) *approximation error is bounded from below by*

$$\begin{aligned} \min_{\{\overline{W}_{\mathcal{T}}\}} \|W_{\mathcal{T}} - \overline{W}_{\mathcal{T}}\|_p &= \min_{\{\overline{W}_{\mathcal{T}}\}} \max_{l \geq 1} \|W_{\mathcal{T}}(l) - \overline{W}_{\mathcal{T}}(l)\|_p \\ &\geq \min_{\{\overline{W}_{\mathcal{T}}\}} \max_{l \geq 1} \left| e^{\frac{1-p}{p} S_p(W_{\mathcal{T}}(l))} \|W_{\mathcal{T}}\|_1 - e^{\frac{1-p}{p} S_p(\overline{W}_{\mathcal{T}}(l))} \|\overline{W}_{\mathcal{T}}\|_1 \right|, \end{aligned} \tag{9}$$

where $S_{\alpha \equiv p}(W(l))$ *is the α-Rényi entropy [Equation (6)].*

Proof. Equation (9) is easily proven by noting the Minkowski inequality $\|A + B\|_p \leq \|A\|_p + \|B\|_p$ and that $(1 - \alpha) S_\alpha(l) = \alpha \log \|W_{\mathcal{T}}(l)\|_\alpha - \alpha \log \|W_{\mathcal{T}}(l)\|_1$ when $\alpha \equiv p \geq 1$ [Equation (6)]. □

The worst-case bound [Equation (9)] is optimized whenever $S_p(\overline{W}_{\mathcal{T}}(l))$ scales the same way as $S_p(W_{\mathcal{T}}(l))$ does. Assuming $S_p(W_{\mathcal{T}}(l)) = C + C' \log l$, then an MPS-type $\overline{W}_{\mathcal{T}}$ cannot efficiently approximate $W_{\mathcal{T}}$ unless D_{II} increases with $\log l$ too, from which the total number of free parameters $\sim PLD_{\text{II}}^2$ [Figure 2b], however, becomes unbounded. By contrast, a MERA-type $\overline{W}_{\mathcal{T}}$ matches the scaling, by which the total number of free parameters $\sim (D^4 + D^3)L$ (where $D \equiv D_{\text{II}}, \cdots = \exp C'$) is efficient enough for any worst-case l.

It is unknown how quantitatively the failure to approximate $W_{\mathcal{T}}$ may impact the expressive power given in Equation (8). The disappearance of the worst-case bound in Equation (9) is a necessary condition for Equation (8) to be valid.

4. Results

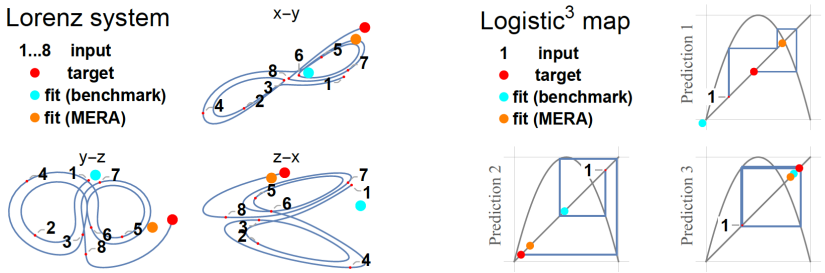
We investigated the accuracy of LSTM-MERA and its generalization ability on different chaotic time series datasets by evaluating the root mean squared error (RMSE) of its one-step-ahead predictions against target values. The benchmark for comparison was chosen to be a vanilla LSTM, of which the hidden dimension h was arbitrarily chosen in advance. LSTM-MERA (and other architectures if present) was built upon the benchmark.

Each time series dataset for training/testing consisted of a set of N_X time series, $\{X^i | i = 1, 2, \cdots, N_X\}$. Each time series $X^i = \{x_t^i | t \in T^i\}$ is of fixed length $|T^i| =$ input steps $+ 1$ so that all but the last step of X^i were inputs, whereas the last step was the one-step-ahead target to be predicted. The dataset $\{X^i\}$ was divided into two subsets—one for testing and one for training, which was further randomly split into a plain training set and a validation set by 80% : 20%. Complete details are given in Appendix C.

All models were trained using Mathematica 12.0 on its NN infrastructure, Apache MXNet, using an ADAM optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-5}$. The learning rate $= 10^{-2}$ and batch size $= 64$ were chosen *a priori*. The NN parameters producing the lowest validation loss during the entire training process were accepted.

4.1. Comparison of LSTM-Based Architectures

When evaluating the advantages of LSTM-MERA, a controlled comparison is essential to confirm that the architecture of LSTM-MERA is inherently better than that of other architectures, not just because the increase of the number of free and learnable parameters (even though more parameters do not necessarily mean more learning power). Here, we studied different architectures (Figure 3) that were all built upon the LSTM benchmark and shared nearly the same number of parameters (param. #). A "wider" LSTM was simply built by increasing h. A "deeper" LSTM was built by stacking two LSTM units as one unit. In particular, LSTM-MPS and LSTM-MERA were built and compared.



(**a**) (**b**)

	h	L	P	$\{D_i\}$	param. #	RMSE	h	L	P	$\{D_i\}$	param. #	RMSE
Benchmark	7	–	–	–	332	0.307	2	–	–	–	35	0.259
"Wider"	11	–	–	–	696	0.279	16	–	–	–	1169	0.187
"Deeper"	7	–	–	–	640	0.105	11	–	–	–	1156	0.204
MPS	7	2^3	2	$\{P,4\}$	663	0.088	2	2^3	2	$\{P,9\}$	1231	0.181
MERA	**7**	$\mathbf{2^3}$	**2**	$\mathbf{\{P,2,3\}}$	**640**	**0.066**	**2**	$\mathbf{2^3}$	**2**	$\mathbf{\{P,4,4\}}$	**1053**	**0.010**

Figure 3. Comparison of different LSTM-based architectures. (**a**) The Lorenz system is a three-dimensional continuous-time dynamical system notable for its chaotic behavior. Discretization: $\Delta t = 0.5$. Input steps = 8, training : validation : test = 2400 : 600 : 2000, and number of epochs = 120 for all models. (**b**) Logistic "cubed" map, i.e., a logistic map re-sampled every three steps. Input steps = 1, training : validation : test = 8000 : 2000 : 500, and number of epochs = 200 for all models. Note that unlike continuous-time dynamical systems, chaos in discrete maps is generally harder to learn.

4.1.1. Lorenz System

Figure 3a describes the forecasting task on the Lorenz system and shows the training results of the LSTM-based models. $\Delta t = 0.5$ was chosen for discretization, which was large enough that the resultant time series hardly exhibited any pattern without the help of a phase line [Figure 3a, input 1–8].

In general, non-tensorized LSTM models performed worse than tensorized LSTM models. After the number of free parameters increased from 332 (benchmark) to 668 ± 28, both the "wider" and "deeper" LSTMs showed signs of overfitting. The "deeper" LSTM yielded lower RMSE than the "wider" LSTM, confirming the common sense that a deep NN is more suitable for generalization than a wide NN [45]. Both LSTM-MPS and LSTM-MERA yielded better RMSE and showed no sign of overfitting. However, LSTM-MERA was more powerful, showing an improvement of ∼25% over LSTM-MPS in RMSE [Figure 3a].

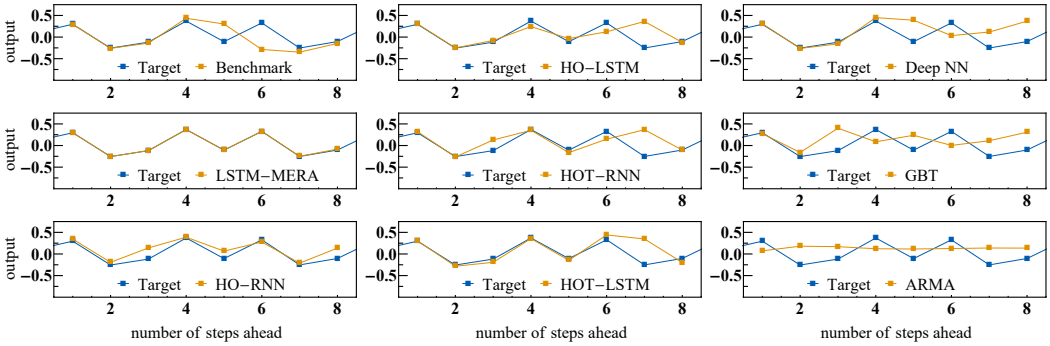
4.1.2. Logistic Map

Figure 3b describes a specific forecasting task on the simplest one-dimensional discrete-time map—the *logistic map*: predicting the target given only a three-step-behind input. Different LSTM models yielded very different results when learning this complex task. After the number of free parameters increased from 35 (benchmark) to 1142 ± 89, all LSTM models yielded lower RMSE than the benchmark. Only LSTM-MERA was able to reach a much lower RMSE (presumably a global minimum) with a remarkable improvement of ∼94% over LSTM-MPS [Figure 3b]. We infer that the local minima reached by the other LSTM models might correspond to the infinite numbers of unstable quasi-periodic cycles in the chaotic phases. In fact, as shown in Figure 3b, Prediction 3, the benchmark fit the target better than LSTM-MERA for this specific example of a quasi-period-2 cycle. However, LSTM-MERA learned the full chaotic behavior and thus performed much better on general examples.

The learning process for the logistic map task was indeed very random, and different realizations yielded very different results. In many realizations, non-tensorized LSTM models did not even learn any patterns at all. By contrast, tensorized LSTM models were more stable in learning.

4.2. Comparison with Statistical/ML Models

We compared LSTM-MERA with more general models, including traditional statisical and ML models, including RNN-based architectures (Figure 4). Specifically, we looked into HOT-RNN/LSTM, which is also claimed to be able to learn chaotic dynamics (e.g., the Lorenz system) through tensorization [28]. Furthermore, for each model we fed its one-step-ahead predictions back so as to make predictions for the second step, and kept feeding back and so on. In theory, the prediction error at the t-th step should increase exponentially with t for chaotic dynamics [Equation (1)].



Model	param. #	RMSE ($\times 10^{-2}$)		
		1 step	2 steps	4 steps
Benchmark	35	1.54	7.63	32.03
LSTM-MERA	**89**	**0.19**	**0.89**	**13.77**
HO-RNN	23	11.91	23.16	27.69
HO-LSTM	83	2.96	14.50	47.99
HOT-RNN	81	12.04	23.76	29.61
HOT-LSTM	315	1.39	6.40	26.83
Deep NN	17950	0.81	3.66	23.49
GBT	–	3.15	16.25	31.37
ARMA	–	24.95	24.93	23.54

Figure 4. Comparison of different statistical/ML models on Gauss "cubed" map, i.e., a Gauss iterated map re-sampled every three steps. Note that the Gauss iterated map is a one-dimensional chaotic map of which the dynamics is smoother than the logistic map and should be easier to learn. Input steps $= 8$. For RNN-based models, $h = 2$, $L = 2^2$, $P = 2$, $\{D_{\mathrm{I}}, D_{\mathrm{II}}, \cdots\} = \{P, 2\}$, training : validation : test $= 8000 : 2000 : 500$, and number of epochs $= 200$. The explicit "history" length used in HO-RNN/LSTM [15] and HOT-RNN/LSTM [28] is also L, and the tensor-train ranks are all D_{II}. Deep NN: depth $= 8$ (= input steps). GBT: maximum depth $= 8$. ARMA family: ARMA$(3, 4)$.

Gauss Iterated Map

We tested the one-step-ahead learning task on the *Gauss "cubed" map* on plain HO-RNN/LSTM [15] and its tensorized version HOT-RNN/LSTM [28]. The explicit "history" length was chosen to be equal to our physical Length L. The tensor-train ranks were all chosen to be equal to D_{II}, as when we built the MPS structure in LSTM-MPS.

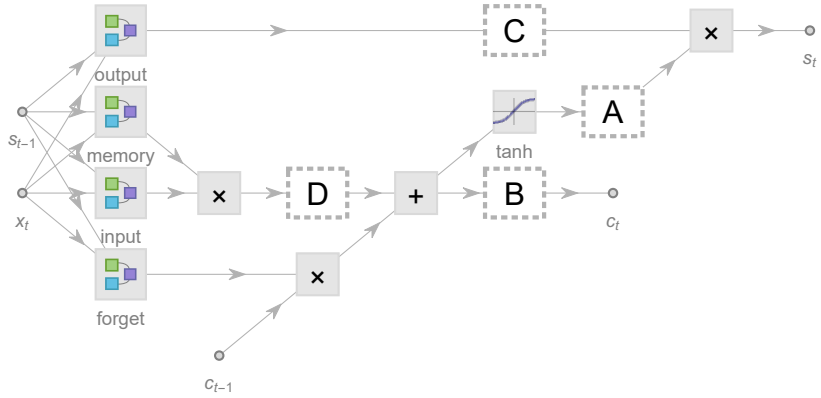
Figure 4 shows that neither HO-RNN nor HO-LSTM performed better than the benchmark, suggesting that introducing explicit non-Markovian dependence (Appendix B.1) is not helpful for capturing chaotic dynamics where the existing nonlinear complexity is never long-term. HOT-LSTM was better than the benchmark because of its MPS structure, suggesting that tensorization, on the other hand, is indeed helpful for forecasting chaos. LSTM-MERA was still the best, with an improvement of ∼88% over the benchmark. Interestingly, the benchmark itself as a vanilla LSTM was already much better than plain RNN architectures (HO-/HOT-RNN).

The learning task was next tested on fully connected deep NN architectures of depth ≤ 8 (equal to the input steps). At each depth three units were connected in series: a linear layer, a scaled exponential linear unit, and a dropout layer. Hyperparameters were determined by means of an optimal search. The best model having the lowest validation loss consisted of $17,950$ free parameters. The task was also tested on GBT of maximum depth $= 8$, as well as on the ARMA family (ARMA, ARIMA, FARIMA, and SARIMA), among which the best statistical model selected by Kalman filtering was ARMA$(3,4)$.

With enough parameters, the deep NN became the second best (Figure 4). All RMSE values increased when making longer-step-ahead predictions, and for the four-step-ahead task the deep NN and LSTM-MERA were the only models that did not overfit and still performed better than the statistical model, ARMA, which made no learning progress but only trivial predictions.

4.3. Comparison with LSTM-MERA Alternatives

Here we tested the ability of LSTM-MERA in the learning of short-term nonlinear complexity by changing its NN topology (Figure 5). We expected to see that, to achieve the best performance, our tensorization (dashed rectangle in Figure 1) should indeed act on the state propagation path $c_t \to s_t$, not on $s_{t-1} \to s_t$ or $c_{t-1} \to c_t$.



Thomas' cyclically symmetric dynamical system		
Model	Site	RMSE ($\times 10^{-1}$)
Benchmark		1.13
LSTM-MERA	**A**	**0.45**
Alternatives	B	1.12
	C	1.10
	D	0.73

Figure 5. Comparison of LSTM-MERA (where the additional layers from Figure 1 are located at Site A) with its alternatives (where the additional layers are instead located at Sites B, C or D), tested on Thomas' cyclically symmetric system, a three-dimensional chaotic dynamical system known for its cyclic symmetry $\mathbf{Z}/3\mathbf{Z}$ under change of axes. Discretization: $\Delta t = 1.0$. Input steps $= 8$, $h = 4$, $L = 2^4$, $P = 4$, $\{D_{\mathrm{I}}, D_{\mathrm{II}}, \cdots\} = \{P, 2, 2, 4\}$, training : validation : test $= 2400 : 600 : 2000$, and number of epochs $= 40$.

Thomas' Cyclically Symmetric System

We investigated different LSTM-MERA alternatives on Thomas' cyclically symmetric system (Figure 5) in order to see if the short-term complexity could still be efficiently learned. The embedded layers, in addition to being located at Site A (the proper NN topology of LSTM-MERA), were also located alternatively at Site B, C or D for comparison. The benchmark was a vanilla LSTM with no embedded layers.

As expected, the lowest RMSE was produced by the proper LSTM-MERA but not its alternatives (Figure 5). The improvement of the proper LSTM-MERA over the benchmark was ∼60%. Interestingly, two alternatives (Site B, Site C) performed *barely* better than the benchmark even with more free learnable parameters. In fact, in the case in which the state propagation path $c_{t-1} \to c_t$ is tensorized (Site B), the long-term gradient propagation along cell states is interfered and the performance of LSTM is deterred; when the path $s_{t-1} \to s_t$ is tensorized (Site C), the improvement is the same as just on a plain RNN and is thus also limited. Hence, proper LSTM-MERA NN topology is critical for improving the performance of learning short-term complexity.

4.4. Generalization and Parameter Dependence of LSTM-MERA

The inherent advantage of LSTM-MERA and its ability to learn chaos have been shown. Hereafter investigated are its parameter dependence, as well as its generalization ability (Figure 6). Each following model (benchmark versus LSTM-MERA) was sufficiently trained through the same number of epochs so that it could reach the lowest stable RMSE. In-between check points were chosen during training, in which models were tested a posteriori on the test data to confirm that an RMSE minimum had eventually been reached.

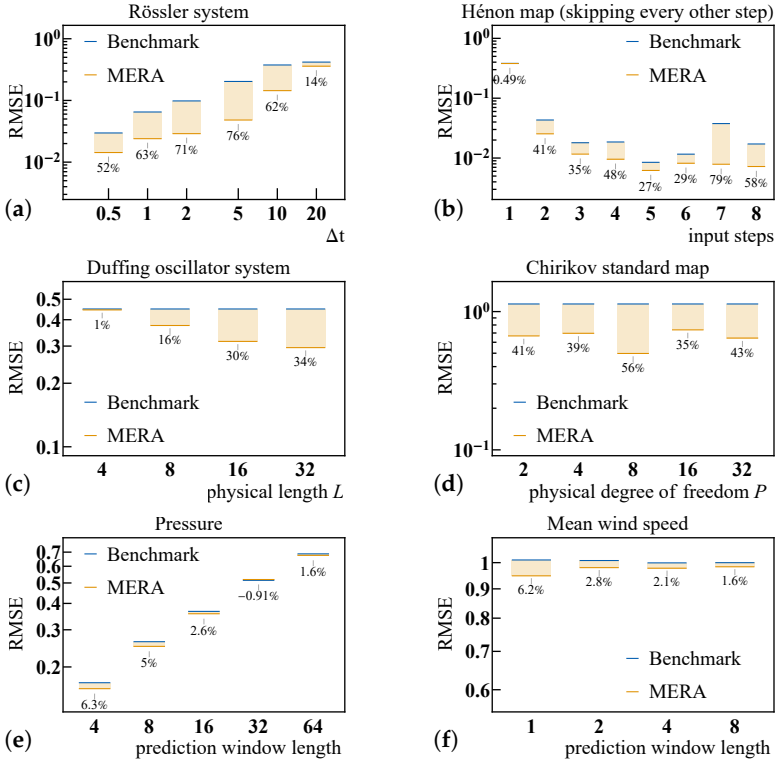


Figure 6. Generalization and parameter dependence of LSTM-MERA. (**a**) Rössler system, another three-dimensional chaotic dynamical system similar to the Lorenz system. Discretization: varying Δt. Input steps $= 4$, $h = 4$, $L = 2^4$, $P = 2$, and $\{D_{\mathrm{I}}, D_{\mathrm{II}}, \cdots\} = \{P, 2, 2, 4\}$. (**b**) One-dimensional, second-order Hénon map, re-sampled by skipping every other step. $h = 4$, $L = 2^3$, $P = 2$, and $\{D_{\mathrm{I}}, D_{\mathrm{II}}, \cdots\} = \{P, 2, 4\}$, whereas the input steps varied. (**c**) Duffing oscillator system. Discretization: $\Delta t = 10.0$. Input steps $= 8$, $h = 4$, $P = 4$, and $\{D_{\mathrm{I}}, D_{\mathrm{II}}, \cdots\} = \{P, 3, 3, \cdots\}$ of which the length varies with L. (**d**) Chirikov standard map. Input steps $= 2$, $h = 2$, $L = 2^3$, and $\{D_{\mathrm{I}}, D_{\mathrm{II}}, \cdots\} = \{P, 4, 4\}$ where P varies. (**e**) Pressure, sampled every eight minutes. Input steps $= 16$, $h = 4$, $L = 2^4$, $P = 4$, $\{D_{\mathrm{I}}, D_{\mathrm{II}}, \cdots\} = \{P, 2, 2, 4\}$, and training : validation : test $= 6400 : 1600 : (\gtrsim 13{,}800)$. (**f**) Mean wind speed, daily sampled. Input steps $= 16$, $h = 4$, $L = 2^3$, $P = 4$, $\{D_{\mathrm{I}}, D_{\mathrm{II}}, \cdots\} = \{P, 2, 2\}$, and training : validation : test $= 128 : 32 : (\gtrsim 7000)$.

4.4.1. Rössler System

In theory, a chaotic time series of larger Δt should be harder to learn [Equation (1)]. This is confirmed in Figure 6a, in which a larger Δt corresponds to a larger RMSE for both models. The greatest improvement of LSTM-MERA over the benchmark was ∼76%, observed at $\Delta t = 5$. The improvement was less when Δt increased, possibly because the time series became too random to preserve any feasible pattern even for LSTM-MERA. The improvement was also less when Δt was small, as the time series was smooth enough and the first-order (linear) time-dependence predominated, which a vanilla LSTM could also learn.

4.4.2. Hénon Map

In view of the fact that the time-dependence is second-order [Figure 6b], there was no explicit and exact dependence between the input and target in the time series dataset. Different input steps were chosen for comparison. When input steps $= 1$, there was no sufficient information to be learned other than a linear dependence between the input and target, and thus both the benchmark and LSTM-MERA performed the same [Figure 6b]. When input steps > 1, however, the time-dependence could be learned implicitly and "bidirectionally" given enough history in length. LSTM-MERA constantly exhibited an average improvement of 45.3%, the fluctuation of which was mostly due to the learning instability not of LSTM-MERA but of the benchmark.

4.4.3. Duffing Oscillator System

Based on Figure 6d, it was clearly observed that larger L yielded better RMSE values. The improvement related to L was significant. This result is not unexpected, since L determines the depth of the MERA structure, with a larger depth corresponding to better generalization ability.

4.4.4. Chirikov Standard Map

As Figure 6d shows, by choosing different P, the greatest improvement of LSTM-MERA over the benchmark was ∼56%, observed at $P = 8$. In general, there was no strong dependence on P.

4.4.5. Real-World Data: Weather Forecasting

The advantage of LSTM-MERA was also tested on real-world weather forecasting tasks [Figure 6e,f]. Unlike for the synthetic time series, here we removed the first-layer translational symmetry [Equation (A1)] previously imposed on LSTM-MERA so that presumed non-stationarity in real-world time series could be better addressed. To perform practical multi-step forecasting, we kept the one-step-ahead prediction architecture of LSTM, yet regrouped the original time series by choosing different prediction window lengths (Appendix C.3).

The improvement of LSTM-MERA over the benchmark was less significant. The average improvement was ∼3.0%, whereas the greatest improvement was ∼6.3%, considering that the prediction window length was small and reflecting that LSTM-MERA is better at capturing short-term nonlinear complexity rather than long-term non-Markovianity. Note that, in the second dataset [Figure 6f], we deliberately used a very small number (=128) of training data to test the overfitting resistibility of the models. Interestingly, LSTM-MERA did not generally perform worse than vanilla LSTM even with more parameters, probably due to the deep architecture of LSTM-MERA.

5. Discussion and Conclusions

The limitations of our model mostly come from the fact that it is only better than traditional LSTM at capturing short-term nonlinearity but not long-term non-Markovianity, and thus its improvement on long-term tasks such as sequence prediction would be limited. That being said, the advantages of tensorizing state propagation in LSTM are evident, including: (1) Tensorization is the most suitable method for the forecasting of nonlinear

chaos since nonlinear terms are treated explicitly and weighted equally by polynomials. (2) Theoretical analysis is conductible since an orthogonal polynomial basis on k-Sobolev space is always available. (3) Tensor decomposition techniques (in particular, from quantum physics) are applicable, which in turn can identify chaos from a different perspective, i.e., tensor complexity (tensor ranks, entropies, etc.).

Our tensorized LSTM model not only offers a general and efficient approach for capturing chaos—as demonstrated by both theoretical analysis and experimental results, showing great potential in unraveling real-world time series—but also brings out a fundamental question of how tensor complexity is related to the learnability of chaos. Our conjecture that a tensor complexity of $S_\alpha(l) = \Theta(\log l)$ in terms of α-Rényi entropy [Equation (6)] generally performs better than $S_\alpha(l) = \Theta(1)$ in chaotic time series forecasting will be further investigated and formalized in the near future.

Author Contributions: Methodology: X.M. and T.Y.; numerical experiments: X.M.; conceptualization: T.Y.; validation: X.M. and T.Y.; formal analysis: X.M.; draft preparation: X.M.; review and editing: X.M. and T.Y.; funding acquisition: X.M. All authors have read and agreed to the published version of the manuscript.

Funding: X.M. was supported by the NetSeed: Seedling Research Award of the Network Science Institute of Northeastern University.

Acknowledgments: The authors would like to thank H. Eugene Stanley (Boston University), Jan Engelbretch (Boston College), Jing Ma (Boston University), Xu Yang (Ohio State University), and Bowen Zhao (Boston University) for helpful discussions.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ML	Machine Learning
NN	Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short Term Memory
HO-	Higher-Order-
HOT-	Higher-Order-Tensorized-
MPS	Matrix Product State
MERA	Multiscale Entanglement Renormalization Ansatz
EE	Entanglement Entropy
DOF	Degree Of Freedom
RMSE	Root Mean Squared Error
ARMA	Auto-Regressive Moving Average
GBT	Gradient Boosted Trees
HMM	Hidden Markov Models

Appendix A. Variants of LSTM-MERA

Appendix A.1. Translational Symmetry

In condensed matter physics, the many-body states studied are usually translational invariant in L, which puts additional constraints on their many-body state structures (MPS, MERA, etc.). Inspired by this, a variant of LSTM-MERA can be constructed by imposing such constraints on the MERA structure too, i.e., by forcing the disentanglers belonging to the same level to be equal to each other. For example, at the first level of the MERA structure [Level I, red in Figure 2d], a constraint

$$[u_{\mathrm{I}}^{(i)}]_{\mu\nu}^{\alpha\beta} \equiv [u_{\mathrm{I}}]_{\mu\nu}^{\alpha\beta}, \quad i = 1, 2, \cdots, L/2 \tag{A1}$$

can be imposed on the weights of the $L/2$ disentanglers. Such a constraint can also be imposed on isometries/inverse isometries, as well as higher levels. When testing LSTM-

MERA on synthetic time series, we have added such a partial translational symmetry constraint on and only on Level I for the purpose of controlling the number of free learnable parameters in our model.

Appendix A.2. Dilational Symmetry

A dilational symmetry constraint is exclusive for MERA, since it has been used in condensed matter physics for representing scaling invariant quantum states. A variant of LSTM-MERA can thus be introduced by imposing the same constraint, i.e., by forcing all disentanglers even from different levels to be equal to each other,

$$\begin{aligned} [u_{\mathrm{I}}^{(i)}]_{\mu\nu}^{\alpha\beta} \equiv [u_{\mathrm{I}}]_{\mu\nu}^{\alpha\beta} \equiv [u_{\mathrm{II}}^{(j)}]_{\mu\nu}^{\alpha\beta} \equiv [u_{\mathrm{II}}]_{\mu\nu}^{\alpha\beta} \equiv \quad \cdots \quad , \\ i = 1, 2, \cdots, L/2; \quad j = 1, 2, \cdots, L/4; \qquad \cdots \quad , \end{aligned} \tag{A2}$$

as well as isometries. This variant of LSTM-MERA may greatly decrease the number of free learnable parameters but may also reduce expressive power.

Appendix A.3. Normalization/Unitarity

Another subtle fact concerning many-body state structures is that the represented states must be normalized. In fact, normalization layers are also widely used in NN architectures, especially for deep NNs, in which the training may suffer from the vanishing gradient problem. In light of this, we have added normalization layers between different LSTM-MERA layers. No extra freedom has been introduced because the "norm" is already a degree of freedom implicitly given by the weights of the disentanglers/isometries.

Similarly, the unitarity of the disentanglers [36] is no longer required. The additional degrees of freedom do not affect the essential MERA structure but may significantly speed up our training.

Appendix B. Common LSTM Architectures

Appendix B.1. HO- and HOT-RNN/LSTM

HO-RNN/LSTM [15] were first introduced to address the problem of explicitly capturing long-term dependence, by changing all gates in LSTM [Equation (2)] into

$$g(x_{t-1}, 1 \oplus s_{t-1} \oplus s_{t-2} \oplus \cdots \oplus s_{t-L}; \mathcal{W}). \tag{A3}$$

Equation (A3) only includes linear (first-order polynomial) terms. As higher-order improvements, HOT-RNN/LSTM [28] were later introduced to include nonlinear (higher-order polynomial) terms as tensor products for the entire non-Markovian dependence,

$$g(x_{t-1}, (1 \oplus s_{t-1} \oplus s_{t-2} \oplus \cdots \oplus s_{t-L})^{\otimes P}; \mathcal{W}). \tag{A4}$$

The weight tensor $\mathcal{W}$ can be further approximated by means of the tensor-train technique [28], which is the same as MPS.

Note that L in Equations (A3) and (A4) is not a virtual dimension but the true time lag. Therefore, to increase the tensorization complexity one has to explicitly increase the time lag dependence. As a comparison, in LSTM-MERA, L is an artificial dimension that can be freely adjusted to reflect the true short-term nonlinear complexity.

Appendix C. Preparation of Time Series Datasets

Appendix C.1. Discrete-Time Maps

Each time series dataset for discrete-time maps was constructed as follows: first, two arrays were produced by the discrete-time map, one with initial conditions (training) and the other one with initial conditions (testing); next, for both arrays, a time window of fixed length (input steps $+1$) moved from the beginning to the end, step by step, and thus extracted a sub-array of length (input steps $+1$) at each step; each extracted sub-array was

a time series. All time series (from both the training array and the testing array) made up the entire time series dataset and served for training and testing, respectively. The initial conditions for training and testing were made different on purpose in order to test the generalization ability of the models, yet they were chosen to belong to the same chaotic regime so that the generality of their subsequent chaotic dynamics was always guaranteed by ergodicity. We investigated four different dynamical systems:

$$\begin{aligned}
&\textbf{Logistic map:} && x_{n+1} = rx_n(1-x_n);\\
&\textbf{Gauss iterated map:} && x_{n+1} = \exp\left(-\alpha x_n^2\right) + \beta;\\
&\textbf{Hénon map:} && x_{n+1} = 1 - ax_n^2 + bx_{n-1};\\
&\textbf{Chirikov standard map:} && p_{n+1} = (p_n + K\sin\theta_n) \bmod 2\pi,\\
& && \theta_{n+1} = (\theta_n + p_{n+1}) \bmod 2\pi.
\end{aligned}$$

Details of the above systems are listed in Table A1.

Table A1. Discrete-time maps in chaotic phases. $\lambda_{1,2}$ are the Lyapunov exponents.

	Logistic	**Gauss**	**Hénon**	**Chirikov**
Dimension	1	1	1	2
Parameters	$r = 4$	$\alpha = 6.2$ $\beta = -0.55$	$a = 1.4$ $b = 0.3$	$K = 2.0$
Initial condition (training)	$x_0 = 0.61$	$x_0 = 0.31$	$x_0 = 0.2$ $x_1 = 0.3$	$p_0 = 0.777$ $\theta_0 = 0.555$
Initial condition (testing)	$x_0 = 0.11$	$x_0 = 0.91$	$x_0 = 0.5$ $x_1 = 0.6$	$p_0 = 0.333$ $\theta_0 = 0.999$
λ_1 λ_2	$\ln 2$ –	0.37 –	0.42 −1.62	0.45 −0.45

Appendix C.2. Continuous-Time Dynamical Systems

Each time series dataset for continuous-time dynamical systems was constructed differently than in Appendix C.1: only one array was produced by discretizing the dynamical system by Δt given the initial conditions; then the array was *standardized*; a time window still moved from the beginning to the end and extracted a sub-array of length (input steps $+1$) at each step; each extracted sub-array was a time series. All time series made up the entire time series dataset, which was then randomly divided into two subsets, one for testing and one for training. Four different dynamics were investigated:

Lorentz system:

$$\frac{dx}{dt} = \sigma(y-x),\ \frac{dy}{dt} = x(\rho - z) - y,\ \frac{dz}{dt} = xy - \beta z;$$

Thomas' cyclically symmetric system:

$$\frac{dx}{dt} = \sin y - bx,\ \frac{dy}{dt} = \sin z - by,\quad \frac{dz}{dt} = \sin x - bz;$$

Rössler system:

$$\frac{dx}{dt} = -y - z,\quad \frac{dy}{dt} = x + ay,\quad \frac{dz}{dt} = b + z(x-c);$$

Duffing oscillator system:

$$\frac{d^2x}{dt^2} + \delta\frac{dx}{dt} + \alpha x + \beta x^3 = \gamma\cos(\omega t).$$

Details of the above systems are listed in Table A2.

Table A2. Continuous-time dynamical systems in chaotic phases. $T_{\max}$ is the maximum solution range, and $\lambda_{1,2,3}$ are the Lyapunov exponents.

	Lorentz	**Thomas**	**Rössler**	**Duffing**
Dimension	3	3	3	1
Parameters	$\rho = 28$ $\sigma = 10.0$ $\beta = 8/3$	$b = 0.1$	$a = 0.1$ $b = 0.1$ $c = 14$	$\alpha = 1.0$ $\beta = 5.0$ $\delta = 0.02$ $\gamma = 8.0$ $\omega = 0.5$
Initial condition	$x_0 = 0$ $y_0 = 1$ $z_0 = 0$	$x_0 = 0$ $y_0 = 1$ $z_0 = 0$	$x_0 = 0$ $y_0 = 1$ $z_0 = 0$	$x_0 = 0$ $\dot{x}_0 = 1$
λ_1 λ_2 λ_3	0.91 0 −14.57	0.06 0 −0.36	0.07 0 −11.7	0.01 0 −0.03
$T_{\max}$	[0, 2500]	[0, 5000]	[0, 100,000]	[0, 50,000]

Appendix C.3. Real-World Time Series: Weather

The data were retrieved using Mathematica's *WeatherData* function, https://reference.wolfram.com/language/note/WeatherDataSourceInformation.html (Figure A1) (accessed on 1 September 2021), and detailed information about the data has been provided in Table A3. Missing data points in the raw time series were reconstructed via linear interpolation. The raw time series was then regrouped by choosing different prediction window lengths, for example, a prediction window length = 4 means that every four consecutive steps in the time series are regrouped together as a one-step four-dimensional vector. Then, the dataset was constructed from the regrouped time series the same way as in Appendix C.2 using a moving window on it after standardization.

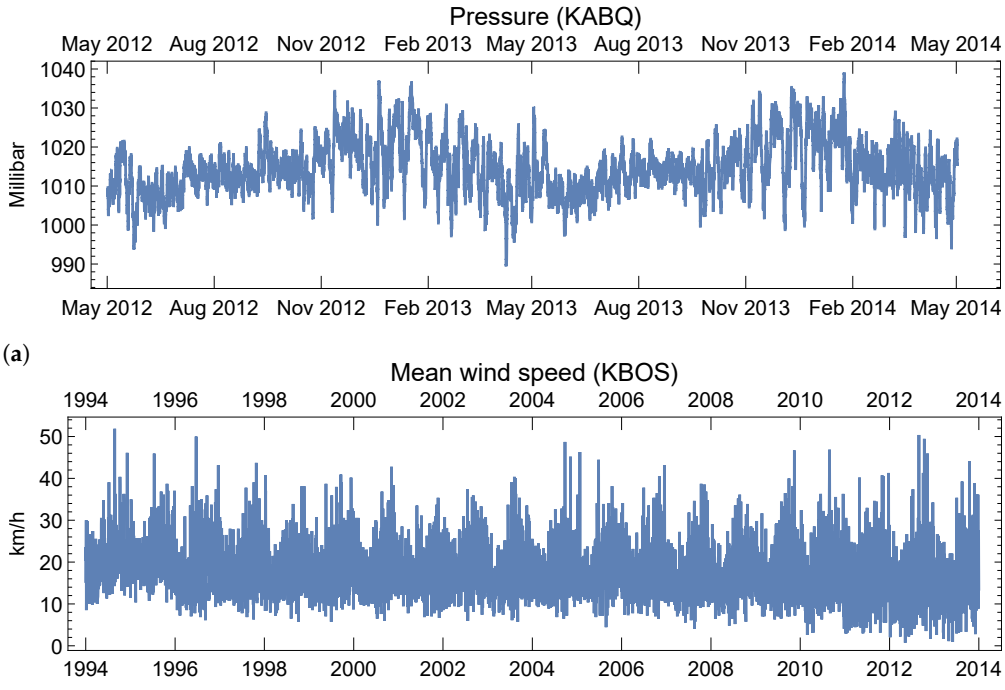


Figure A1. Weather time series. (**a**) Pressure (every eight minutes) before standardization. (**b**) Mean wind speed (daily) before standardization.

Table A3. Details of weather datasets used in the main article.

	Pressure	Mean Wind Speed
Location	ICAO:KABQ	ICAO:KBOS
Span	05/01/2012–05/01/2014	05/01/1994–05/01/2014
Frequency	8 min	1 day
Total length	22,426	7299

References

1. Makridakis, S.; Spiliotis, E.; Assimakopoulos, V. Statistical and machine learning forecasting methods: Concerns and ways forward. *PLoS ONE* **2018**, *13*, e0194889. [CrossRef] [PubMed]
2. Cheng, Y.; Anick, P.; Hong, P.; Xue, N. Temporal relation discovery between events and temporal expressions identified in clinical narrative. *J. Biomed. Inform.* **2013**, *46*, S48–S53. [CrossRef] [PubMed]
3. Box, G.E.P.; Jenkins, G.M.; Reinsel, G.C.; Ljung, G.M. *Time Series Analysis: Forecasting and Control*; Wiley: Hoboken, NJ, USA, 2015.
4. Rabiner, L.; Juang, B. An introduction to hidden Markov models. *IEEE ASSP Mag.* **1986**, *3*, 4–16. [CrossRef]
5. Hong, P.; Huang, T.S. Automatic temporal pattern extraction and association. In Proceedings of the 2002 IEEE International Conference on Acoustics, Speech, and Signal Processing, Orlando, FL, USA, 13–17 May 2002; Volume 2, pp. II–2005–II–2008.
6. Ahmed, N.K.; Atiya, A.F.; Gayar, N.E.; El-Shishiny, H. An empirical comparison of machine learning models for time series forecasting. *Econom. Rev.* **2010**, *29*, 594–621. [CrossRef]
7. Osogami, T.; Kajino, H.; Sekiyama, T. Bidirectional learning for time-series models with hidden units. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 2711–2720.
8. Borovykh, A.; Bohte, S.; Oosterlee, C.W. Conditional time series forecasting with convolutional neural networks. *arXiv* **2018**, arXiv:1703.04691v5.
9. Ding, D.; Zhang, M.; Pan, X.; Yang, M.; He, X. Modeling extreme events in time series prediction. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 1114–1122.
10. Bar-Yam, Y. *Dynamics Of Complex Systems (Studies in Nonlinearity)*; CRC Press: New York, NY, USA, 1999.
11. Jozefowicz, R.; Zaremba, W.; Sutskever, I. An empirical exploration of recurrent network architectures. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 2342–2350.
12. Giles, C.L.; Sun, G.Z.; Chen, H.H.; Lee, Y.C.; Chen, D. Higher order recurrent networks and grammatical inference. In Proceedings of Neural Information Processing Systems, Denver, CO, USA, 27–30 November 1989; pp. 380–387.
13. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
14. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
15. Soltani, R.; Jiang, H. Higher order recurrent neural networks. *arXiv* **2016**, arXiv:1605.00064.
16. Haruna, T.; Nakajima, K. Optimal short-term memory before the edge of chaos in driven random recurrent networks. *Phys. Rev. E* **2019**, *100*, 062312. [CrossRef]
17. Feng, L.; Lai, C.H. Optimal machine intelligence near the edge of chaos. *arXiv* **2019**, arXiv:1909.05176.
18. Kuo, J.M.; Principle, J.C.; de Vries, B. Prediction of chaotic time series using recurrent neural networks. In Proceedings of the Neural Networks for Signal Processing II Proceedings of the 1992 IEEE Workshop, Helsingoer, Denmark, 31 Augest–2 September 1992; pp. 436–443.
19. Zhang, J.S.; Xiao, X.C. Predicting chaotic time series using recurrent neural network. *Chinese Phys. Lett.* **2000**, *17*, 88–90. [CrossRef]
20. Han, M.; Xi, J.; Xu, S.; Yin, F.L. Prediction of chaotic time series based on the recurrent predictor neural network. *IEEE Trans. Signal Process.* **2004**, *52*, 3409–3416. [CrossRef]
21. Ma, Q.L.; Zheng, Q.L.; Peng, H.; Zhong, T.W.; Xu, L.Q. Chaotic time series prediction based on evolving recurrent neural networks. In Proceedings of the 2007 International Conference on Machine Learning and Cybernetics, Hong Kong, China, 19–22 August 2007; Volume 6, pp. 3496–3500.
22. Domino, K. The use of the Hurst exponent to predict changes in trends on the Warsaw stock exchange. *Phys. A* **2011**, *390*, 98–109. [CrossRef]
23. Li, Q.; Lin, R. A new approach for chaotic time series prediction using recurrent neural network. *Math. Probl. Eng.* **2016**, *2016*, 3542898. [CrossRef] [PubMed]
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* **2018**, *474*, 20170844. [CrossRef] [PubMed]
25. Domino, K. Multivariate cumulants in outlier detection for financial data analysis. *Phys. A* **2020**, *558*, 124995. [CrossRef]
26. Yang, Y.; Krompass, D.; Tresp, V. Tensor-train recurrent neural networks for video classification. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; Volume 70, pp. 3891–3900.

27. Schlag, I.; Schmidhuber, J. Learning to reason with third order tensor products. In Proceedings of the Neural Information Processing Systems 2018, Montreal, QC, Canada, 3–8 December 2018; Volume 31, pp. 9981–9993.
28. Yu, R.; Zheng, S.; Anandkumar, A.; Yue, Y. Long-term forecasting using higher order tensor RNNs. *arXiv* **2019**, arXiv:1711.00073v3.
29. Raissi, M. Deep hidden physics models: Deep learning of nonlinear partial differential equations. *J. Mach. Learn. Res.* **2018**, *19*, 1–24.
30. 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.* **2018**, *120*, 024102. [CrossRef]
31. Jiang, J.; Lai, Y.C. Model-free prediction of spatiotemporal dynamical systems with recurrent neural networks: Role of network spectral radius. *Phys. Rev. Res.* **2019**, *1*, 033056. [CrossRef]
32. Qi, D.; Majda, A.J. Using machine learning to predict extreme events in complex systems. *Proc. Natl. Acad. Sci. USA* **2020**, *117*, 52–59. [CrossRef]
33. Graves, A. Generating sequences with recurrent neural networks. *arXiv* **2014**, arXiv:1308.0850v5.
34. Carleo, G.; Cirac, I.; Cranmer, K.; Daudet, L.; Schuld, M.; Tishby, N.; Vogt-Maranto, L.; Zdeborová, L. Machine learning and the physical sciences. *Rev. Mod. Phys.* **2019**, *91*, 045002. [CrossRef]
35. Eisert, J.; Cramer, M.; Plenio, M.B. *Colloquium*: Area laws for the entanglement entropy. *Rev. Mod. Phys.* **2010**, *82*, 277–306. [CrossRef]
36. Vidal, G. Class of quantum many-body states that can be efficiently simulated. *Phys. Rev. Lett.* **2008**, *101*, 110501. [CrossRef]
37. Verstraete, F.; Cirac, J.I. Matrix product states represent ground states faithfully. *Phys. Rev. B* **2006**, *73*, 094423. [CrossRef]
38. Zhang, Y.H. Entanglement entropy of target functions for image classification and convolutional neural network. *arXiv* **2017**, arXiv:1710.05520.
39. Khrulkov, V.; Novikov, A.; Oseledets, I.V. Expressive power of recurrent neural networks. In Proceedings of the 6th International Conference on Learning Representations (ICLR 2018), Vancouver, British Columbia, Canada, 30 April–3 May 2018.
40. Bhatia, A.S.; Saggi, M.K.; Kumar, A.; Jain, S. Matrix product state based quantum classifier. *Neural Comput.* **2019**, *31*, 1499–1517. [CrossRef]
41. Jia, Z.A.; Wei, L.; Wu, Y.C.; Guo, G.C.; Guo, G.P. Entanglement area law for shallow and deep quantum neural network states. *New J. Phys.* **2020**, *22*, 053022. [CrossRef]
42. Bigoni, D.; Engsig-Karup, A.P.; Marzouk, Y.M. Spectral tensor-train decomposition. *SIAM J. Sci. Comput.* **2016**, *38*, A2405–A2439. [CrossRef]
43. Grasedyck, L. Hierarchical singular value decomposition of tensors. *SIAM J. Matrix Anal. Appl.* **2010**, *31*, 2029–2054. [CrossRef]
44. Canuto, C.; Quarteroni, A. Approximation results for orthogonal polynomials in Sobolev spaces. *Math. Comput.* **1982**, *38*, 67–86. [CrossRef]
45. Dehmamy, N.; Barabási, A.L.; Yu, R. Understanding the representation power of graph neural networks in learning graph topology. *arXiv* **2019**, arXiv:1907.05008.