Fractional Tensor Recurrent Unit (fTRU): A Stable Forecasting Model With Long Memory

Hejia Qiu, Chao Li, Ying Weng, Zhun Sun, and Qibin Zhao, *Senior Member, IEEE*

***Abstract*— The tensor recurrent model is a family of nonlinear dynamical systems, of which the recurrence relation consists of a p-fold (called degree-p) tensor product. Despite such models frequently appearing in advanced recurrent neural networks (RNNs), to this date, there are limited studies on their long memory properties and stability in sequence tasks. In this article, we propose a fractional tensor recurrent model, where the tensor degree p is extended from the discrete domain to the continuous domain, so it is effectively learnable from various datasets. Theoretically, we prove that a large degree p is essential to achieve the long memory effect in a tensor recurrent model, yet it could lead to unstable dynamical behaviors. Hence, our new model, named fractional tensor recurrent unit (fTRU), is expected to seek the saddle point between long memory property and model stability during the training. We experimentally show that the proposed model achieves competitive performance with a long memory and stable manners in several forecasting tasks compared to various advanced RNNs.**

***Index Terms*— Long memory, model stability, recurrent unit, tensor degree.**

I. Introduction

RECURRENT neural networks (RNNs) have been popularly used for tasks arising in domains such as time-series analysis and natural language processing. They are powerful because the recurrent dynamics of the hidden states allow the models to remember past information. In the standard RNNs, the transition function of the hidden states obeys an affine transform following an element-wise activation function, and it is known that the vanilla RNN has an inherent difficulty in learning the long-range dependence of the data. As alternatives, a family of RNN variants is proposed, where the affine transition is modified to be higher-degree polynomials [1], [2]. Rich numerical results in existing works demonstrate that the RNNs equipped with polynomial transitions are more expressive than their vanilla counterparts [3], [4].

However, those polynomial-induced variants fail to become mainstream models in sequence tasks. The cause is mainly twofold: theoretically, it remains unclear why the higher-degree models outperform their linear counterparts; empirically, an "over-large" degree would make the model computationally unstable in both the training and inference phases. More problematically, the model size would grow exponentially when the degree increases. Although it can be alleviated by tensor decomposition (TD) [5], the selection of the degree parameters is still challenging due to the expensively exhaustive search [3].

As a consequence, the state-of-the-art raises some critical unresolved questions. *How does the model degree affect the dynamical behavior of the model, such as memory performance and stability? Is it possible to learn the optimal degree from the data in practice to avoid the exhaustive search?* This work aims to shed light on both two questions through theoretical and empirical investigations. The main contributions of this article are summarized as follows.

1) We propose the unified expression of the aforementioned variants of recurrent models named *fractional tensor recurrent unit (fTRU)*, where the degree-p polynomial is reformulated in the fractional domain as a p-fold tensor product of the hidden states multiplication by a weight tensor. The fractional degree-p is learnable during the training of the recurrent model in our design.
2) Theoretically, we prove that a large-degree p is an essential condition to achieve the long memory effect, yet it could lead to unstable dynamical behaviors. The proposed learnable fractional degree is supposed to find the saddle point between long memory and model stability.
3) Hence, our proposed fTRU can provide competitive performance in the recurring tasks compared with other tensor models under static degrees. Various sequence simulations and forecasting experiments are carried out to illustrate both the higher accuracy and stability of the proposed method.

A. Related Works

The earliest study of RNNs with polynomial transition dates back to literature [6]. Later, similar architectures are employed in various machine-learning tasks [1], [7], [8]. On the practical

Manuscript received 12 May 2023; revised 18 September 2023; accepted 22 November 2023. This work was supported in part by Ningbo Major Science and Technology Project under Grant 2022Z126, in part by the University of Nottingham Project under Grant I01200100023, in part by the National Natural Science Foundation of China under Project 62006045, and in part by the JSPS KAKENHI under Project JP20H04249 and Project JP23H03419. *(Hejia Qiu and Chao Li contributed equally to this work.) (Corresponding author: Ying Weng.)*

Hejia Qiu and Ying Weng are with the School of Computer Science, University of Nottingham Ningbo China, Ningbo 315100, China (e-mail: hejia.qiu@nottingham.edu.cn; ying.weng@nottingham.edu.cn).

Chao Li and Qibin Zhao are with RIKEN AIP, Tokyo 103-0027, Japan (e-mail: chao.li@riken.jp; qibin.zhao@riken.jp).

Zhun Sun is with the Graduate School of Information Sciences, Tohoku University, Sendai 980-8577, Japan (e-mail: zhun.sun@tohoku.ac.jp).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TNNLS.2023.3338696.

Digital Object Identifier 10.1109/TNNLS.2023.3338696

side, the polynomials are also applied to constructing the correlation among time steps in multimodal tasks [9]. Compared to those works, our work is the first to investigate the memory property of the model theoretically. The proposed method has the capability of learning the optimal degree parameters from datasets, while in previous works, those parameters are fixed as tuning parameters.

Based on the former work on RNNs, many tensor-based methods have been proposed, mainly including three lines. One line of work is the extension of the polynomial RNNs to higher-degree ones. In the existing works, the closest one to ours is higher-order tensor RNN (HOTRNN) [3], and a similar idea is extended in recent works on convolutional long short-term memory (LSTM) [4]. Compared to those methods, we mainly focus on the issues of how the "tensor order" (i.e., the "degree" mentioned in this article) influences the performance of RNNs and how to obtain the optimal tensor order in a more efficient manner, rather than by the exhaustive search. The second line is to apply TD to compressing RNNs [5], [10]. Unlike ours, they still model the transition function as a "linear" transform, yet reshaping the weights into high-order tensors. The third line of work is to use the tensor networks to analyze the expressive power of RNNs [11]. In contrast, our work pays more effort to the memory property focusing on the impact of the "tensor-order" rather than different decomposition models.

Many former works have been proposed on the long memory property of RNNs [12]. In this work, the theoretical aspect is partially inspired by recent studies from the stochastic and stability perspective [13], [14]. Compared to those works, we focus on more specific nonlinear dynamics of the transition and analyze how the model degree influences the memory property.

The study of multivariate fractional polynomial fitting originates in statistics [15] and is also discussed in the deep-learning community [16]. Our former work [17] is connected to these works but focuses on the memory mechanism. Most recently, we also extended a similar idea into a more general form named fractional tensor network [18], which focuses on different tensor structures and feedforward learning models. In this work, our attention mainly concentrates on fTRU's performance and its stable manners with comprehensive analysis.

B. Structure of This Article

In the following sections, we first explain the related preliminaries, including recurrent network process (RNP) and tensor recurrent unit (TRU); second, address TRU's memory mechanism from the different perspectives; third, propose a fractional degree-learnable approach; finally, conduct experiments for further comparison and analysis.

II. Preliminaries

In this section, we first present basic notations of tensor algebra. After that, we review the definition of long memory in statistics using the terms from RNP [13]. Last, we introduce the TRU.

We define a *tensor* as a multidimensional array of real numbers. To distinguish from the notion of "order" in time-series analysis, we use the term *degree of a tensor* to denote the number of indices. In the rest of the article, we use italic letters to denote scalars, for example, $n, N \in \mathbb{R}$, and use boldface letters to denote vectors and matrices, for example, $\mathbf{h}, \mathbf{y} \in \mathbb{R}^n$ and $\mathbf{W} \in \mathbb{R}^{n\times n}$. For higher-degree tensors, we denote them by calligraphic letters, for example, $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_d}$. Sometimes, we also use calligraphic letters to represent vectors or matrices without ambiguity. The *symmetry* of a tensor is defined as the invariability under arbitrarily reshuffling a subcollection of the tensor's indices, and the *index-shift* operation of a tensor is defined as rotating permuting the indices of a tensor in counterclockwise order. Examples of "symmetry" and "index-shifting" are given in the supplementary material. For an integer k, we use $[k]$ to denote the set of integers from 1 to k. For a real number x, we use $|x|$ and $\operatorname{sgn}(x)$ to denote its absolute value and sign, respectively. If $\mathbf{x} \in \mathbb{R}^{I_x}$ and $\mathbf{y} \in \mathbb{R}^{I_y}$, we use $\langle \mathbf{x}, \mathbf{y}\rangle \in \mathbb{R}$ ($I_x = I_y$ in this case) and $\mathbf{x} \otimes \mathbf{y} \in \mathbb{R}^{I_x\times I_y}$ to, respectively, denote the inner and tensor product between vectors, and their straightforward extension to matrices and tensors. Moreover, we use $\mathbf{x}^{\otimes p} \in \mathbb{R}^{I_x^p}$ to denote the degree-p *tensor power*, which represents the p-fold tensor product of $\mathbf{x}$ with itself. For a degree-q tensor $\mathcal{G} \in \mathbb{R}^{n^q}$ and a vector $\mathbf{x} \in \mathbb{R}^n$, we use $\mathcal{G} \times_i \mathbf{x}, i \in [q]$ to denote the *tensor–vector* product along the ith index. The spectral norm of a tensor $\mathcal{G} \in \mathbb{R}^{I_1\times\cdots\times I_q}$ is denoted by $\|\mathcal{G}\|_2$, which is defined as

$$\|\mathcal{G}\|_2 = \sup_{\mathbf{u}_i\in\mathbb{R}^{I_i}, i\in[q]} \left\|\mathcal{G} \times_1 \mathbf{u}_1 \times_2 \cdots \times_q \mathbf{u}_q\right\|_2 \quad \text{s.t. } \|\mathbf{u}_i\|_2 \le 1 \quad \forall i \tag{1}$$

where $\|\mathbf{u}_i\|_2, \forall i$ denote the Euclidean norm of $\mathbf{u}_i \in \mathbb{R}^n$.

A. Recurrent Network Process

The notion of *RNP* (2) is established to analyze the memory mechanism of RNNs. Specifically, assume a typical RNN that receives input $\{\mathbf{x}^{(t)}\}$, outputs $\{\mathbf{y}^{(t)}\}$, and updates its hidden states $\{\mathbf{h}^{(t)}\}$, where $\mathbf{y}^{(t)} \in \mathbb{R}^l$ and $\mathbf{h}^{(t)} \in \mathbb{R}^m$, then its RNP is defined as a homogeneous Markov chain with a transition function M

$$\begin{pmatrix}\mathbf{y}^{(t)}\\ \mathbf{h}^{(t)}\end{pmatrix} = M\begin{pmatrix}\mathbf{y}^{(t-1)}\\ \mathbf{h}^{(t-1)}\end{pmatrix} + \begin{pmatrix}\varepsilon^{(t)}\\ 0\end{pmatrix} \tag{2}$$

where $\{\varepsilon^{(t)}\}$ represents a sequence of independent and identically distributed (*i.i.d.*) random vectors. Equation (2) can be used to describe various RNNs like the vanilla RNN and LSTM without exogenous inputs. The term *short or long memory* of the stochastic process generated by (2) is roughly defined as below. More precise definition is given in [13] and [19].

Definition 1 (Memory of RNPs, Roughly): The process $\{\mathbf{s}^{(t)} := (\mathbf{y}^{(t),\top}, \mathbf{h}^{(t),\top})^\top \in \mathbb{R}^{l+m}\}$ generated by (2) has short memory if its autocovariance function is summable. Otherwise, the process has a long memory.

The relationship between the memory property and the operator norm of the transition function M is revealed below.

Assumption 1: 1) The joint density function of $\varepsilon(t)$ is continuous and positive everywhere; 2) for some $\kappa \geq 2$, $\mathbb{E}\|\varepsilon^{(t)}\|_2^{\kappa} < \infty$.

Theorem 1 (Given by [13]): Under Assumption 1, if there exist real numbers $0 < a < 1$ and b such that $\|M(\mathbf{x})\|_2 \leq a\|\mathbf{x}\|_2 + b$, then the RNP (2) has short memory.

Definition 1 and Theorem 1 will be used in Section III-A to model and analyze the memory property of TRU. We use them to bridge the gap from the memory property to the tensor power operation.

B. Tensor Recurrent Unit

A *degree-p* TRU, with its input $\mathbf{x}^{(t)} \in \mathbb{R}^l$ and hidden state $\mathbf{h}^{(t)} \in \mathbb{R}^m$ at the time step t is defined as follows[1]:

$$\begin{aligned}\mathbf{h}^{(t)} &= \mathcal{G} \times_1 \begin{pmatrix}\mathbf{x}^{(t)}\\ \mathbf{h}^{(t-1)}\end{pmatrix} \times_2 \cdots \times_p \begin{pmatrix}\mathbf{x}^{(t)}\\ \mathbf{h}^{(t-1)}\end{pmatrix}\\ &= \mathcal{G} \cdot \begin{pmatrix}\mathbf{x}^{(t)}\\ \mathbf{h}^{(t-1)}\end{pmatrix}^{\otimes p}\end{aligned} \tag{3}$$

where $\mathcal{G} \in \mathbb{R}^{n^p \times m}$, $n = l + m$ denotes a degree-$(p+1)$ tensor, which is partially symmetric involving the first p indices. The symbol $\cdot$ is used as a simplified representation of the sequential tensor–vector multiplication in (3). In this case, the current state $\mathbf{h}^{(t)}$ is updated by a system of degree-p homogeneous polynomials of the concatenation of inputs and the historic values of the states. Its more general extensions that include *inhomogeneous* terms can be trivially obtained by padding additional constant values along $\mathbf{x}^{(t)}$. Although activation functions such as "tanh" or "ReLU" are popularly used in the model, in this work, we consider the case that the activation functions are ignored to simplify the theoretical results. We claim that in practical tasks *the nontrivial tensor power operation has the capability of providing sufficient nonlinearity to the model.*

The model (3) would degenerate into the most common linear form when $p = 1$. Under the setting of $p = 2$, (3) can fully describe the dynamics of models in [1] and [20]. For the higher-degree case, (3) is strongly related to various tensor recurrent models [3], [4]. It is worth noting that in those methods, the non-Markovian model is also applied by considering longer historic hidden states in the recurrent process. Our empirical results show that additional historic states are not necessary for the long memory property.

III. Memory of TRU

In this section, we theoretically investigate the memory mechanism of TRU, that is, (3) from both stochastic and stability perspectives and focus on proving how the degree determines the memory property of the model. Note that, in the theoretical analysis, we do not apply TD to the weights, which is different from the works [3], [4]. TD will be exploited in Section IV to develop a degree-learnable model. The proofs can be found in the supplementary material.

A. Perspective From RNPs

Assume that the output of TRU is also obtained by the similar degree-p tensor power form to (3) and there is no exogenous input, that is, $\mathbf{x}^{(t)} = \mathbf{y}^{(t-1)}$, then (2) can be specified as

$$\begin{aligned}\mathbf{s}^{(t)} &= \mathcal{M} \times_1 \mathbf{s}^{(t-1)} \times_2 \cdots \times_p \mathbf{s}^{(t-1)} + \mathbf{e}^{(t)}\\ &= \mathcal{M} \cdot \left(\mathbf{s}^{(t-1)}\right)^{\otimes p} + \mathbf{e}^{(t)} \quad \forall t\end{aligned} \tag{4}$$

where $\mathbf{s}^{(t)} := (\mathbf{y}^{(t),\top}, \mathbf{h}^{(t),\top})^{\top} \in \mathbb{R}^{l+m}, n = l + m$, $\mathcal{M} \in \mathbb{R}^{n^{(p+1)}}$ denotes a degree-$(p+1)$ tensor with a symmetric structure involving the first p indices, and $\mathbf{e}^{(t)} := (\varepsilon^{(t),\top}, 0)^{\top}$. We refer to the stochastic process generated by the model (4) as *tensor RNP (T-RNP)*. Below, we investigate how the degree p influences the memory of T-RNP under Definition 1. First, we show that T-RNP has short memory if the spectral norm of $\mathcal{M}$ is upper-bounded.

Lemma 1: Under Assumption 1, the T-RNP (4) has short memory under Definition 1 if the spectral norm of the tensor $\mathcal{M}$ obeys $\|\mathcal{M}\|_2 < 1$.

Lemma 1 is a natural corollary from Theorem 1. It is implied from the claim that T-RNP has the short memory property if the spectral norm of the tensor $\mathcal{M}$ is sufficiently small. As pointed out in [13], the condition is often satisfied in practice when $p = 1$ (i.e., the linear RNN) because the entries of $\mathcal{M}$ are practically bounded away from one. However, *can we expect that the short memory of T-RNP (4) would be preserved if increasing the degree parameter $p > 1$?* To answer the question, we specify that the entries of $\mathcal{M}$ obey the sub-Gaussian distribution.

Assumption 2 (Sub-Gaussian and Decoupling): The tensor $\mathcal{M}$ is obtained by the average over all p first-p-indices-shifted variants of a tensor $\mathcal{A} \in \mathbb{R}^{n^{(p+1)}}$, of which each entries $\mathcal{A}_{i_1,i_2,\ldots,i_{p+1}}$ is independent, zero-mean, and satisfies $\mathbb{E}(e^{t\mathcal{A}_{i_1,i_2,\ldots,i_{p+1}}}) \leq e^{\sigma^2 t^2/2}$.

It is easily known that the averaged tensor $\mathcal{M}$ is symmetric involving the first p indices. Moreover, since the sub-Gaussian distribution generates samples concentrating around zero with few outliers, the assumption is empirically reasonable as aforementioned. Next, we theoretically reveal how the degree impacts the model's memory property. Suppose that our T-RNP satisfies Assumptions 1 and 2, then we show below that the long memory requires a high model degree. Specifically, we have the following theorem.

Theorem 2 (Long Memory Requires a High Model Degree): Under Assumptions 1 and 2, with high probability, that is, at least $1 - \delta$, if T-RNP (4) has the long memory under Definition 1, then the following inequality obeys:

$$p \geq \frac{p_0}{2}\left(1 + \sqrt{1 + \frac{C_1}{n\sigma^2} - \frac{C_2}{n}}\right) - 1 \tag{5}$$

where $p_0 = \log(3/2)$, $C_1 = 1/(2\log(3/2))$ and $C_2 = 4\log(2/\delta - 3/2)$. More details and proof can be found in the Appendix.

The inequality (5) is obtained using Lemma 2 and the results on the nonasymptotic bound of tensor spectral norm given by [21]. We can see that the degree p is controlled by both the dimension of the model and the distribution of $\mathcal{M}$. Fixing the dimension n, a smaller value of σ^2 leads to a larger value of the degree p for the long memory.

[1]Here, the bias term is omitted for brevity.

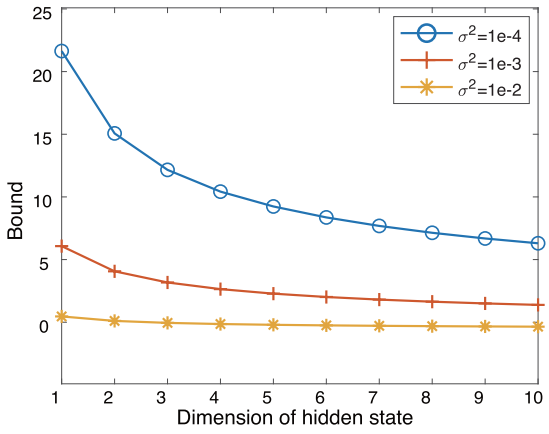


Fig. 1. Illustration to the bound given in Theorem 2, in which $\delta = e^{-9}$.

Theorem 2 shows that the long memory property of TRU requires a sufficiently large degree parameter p, otherwise the model only has a short memory. In Fig. 1, we visualize the curve of the bound given in Theorem 2 under different σ and hidden state dimensions. As shown, to obtain the long memory property, the TRU requires a higher degree of decreasing the variance σ^2. As discussed, the weights of a well-trained RNN are generally far away from 1. In this case, the variance σ^2 would be quite small under the empirical assumption. It implies that a high model degree is necessary to obtain a long memory. On the other side, we can see that the bound decreases when increasing the dimension of the hidden state. It is because the spectral norm of the weight tensor becomes more easily larger than 1 if fixing the distribution yet increasing the dimension. This fact partially reflects why a tensor recurrent model with higher order hidden states can "remember" more information from data, yielding lower training loss.

Intuitively, the model with a higher degree would result in an unbounded nonlinear transition function. In this case, the transition plays a role like an *"amplifier,"* such that at each time step, both the hidden state and input would be amplified by the degree p. As a result, it would become uneasy for the network to "forget" the information a long time ago, that is, the long memory effect.

B. Perspective From Stability Analysis

The long memory effect does not guarantee that the recurrent model is trainable. Therefore, it is important to state whether an unstable behavior would happen in TRU. In particular, *how does the model degree affect the stability of the model?*

Recall (3). To answer the question, we define its stability following the one given in [14] for an arbitrary recurrent model.

Definition 2 (Stability of TRU): The model (3) is stable if there exist some $\lambda < 1$ such that, for any states $\mathbf{h}, \mathbf{h}'$, and input $\mathbf{x}$

$$\left\| \mathcal{G} \cdot \left(\begin{pmatrix} \mathbf{x} \\ \mathbf{h} \end{pmatrix}^{\otimes p} - \begin{pmatrix} \mathbf{x} \\ \mathbf{h}' \end{pmatrix}^{\otimes p} \right) \right\|_2 \leq \lambda \left\| \mathbf{h} - \mathbf{h}' \right\|_2. \tag{6}$$

As shown in Definition. 2, TRU is stable if (3) is λ-Lipschitz with $\lambda < 1$. Hence, we next prove that *the TRU is not Lipschitz continuous for* $p > 1$. Before the main claim, we first give a general form of the Jacobian of (3).

Lemma 2 (Jacobian of the Model): For any tensor $\mathcal{G} \in \mathbb{R}^{n^p \times m}$ of degree-$(p+1)$, $p > 0$, the Jacobian matrix $(\partial \mathbf{h}^{(i)})/(\partial \mathbf{h}^{(i-1)})$ with respect to (3) is equal to

$$J\left(\mathbf{h}^{(i-1)}; \mathbf{x}^{(i)}\right) = \frac{\partial \mathbf{h}^{(i)}}{\partial \mathbf{h}^{(i-1)}} = \sum_{k=1}^{p} \left(\mathcal{G} \cdot \begin{pmatrix} \mathbf{x}^{(i)} \\ \mathbf{h}^{(i-1)} \end{pmatrix}^{\otimes p/\{k\}} \right) \times_{p+1} \begin{pmatrix} \mathbf{0}_l \\ \mathbf{I}_m \end{pmatrix} \tag{7}$$

where $\mathbf{0}_l \in \mathbb{R}^{l \times m}$ denotes a matrix filled by zeros, $\mathbf{I}_m \in \mathbb{R}^{m \times m}$ is an identity matrix, and the operator $(\cdot)^{\otimes p/\{k\}}$ denotes the sequential "tensor–vector" product along the indices in the ordered set $[p]/\{k\}$. If $\mathcal{G}$ is symmetric among the first p indices, then (7) can be simplified as

$$J_s\left(\mathbf{h}^{(i-1)}; \mathbf{x}^{(i)}\right) = p \left(\mathcal{G} \cdot \begin{pmatrix} \mathbf{x}^{(t)} \\ \mathbf{h}^{(t-1)} \end{pmatrix}^{\otimes (p-1)} \right) \times_{p+1} \begin{pmatrix} \mathbf{0}_l \\ \mathbf{I}_m \end{pmatrix}. \tag{8}$$

As shown in Lemma 2, the Jacobian is a constant matrix when $p = 1$. It implies that a linear recurrent model is stable if the spectral norm of $\mathcal{G}$ is sufficiently small. In this case, the model has the short memory as Lemma 1. Next, we give the main claim by proving that the Jacobian (7) would be unbounded, which implies the high-degree TRU would be in an unstable regime.

Theorem 3 (High Degree Models Lead to Unstable Behaviors): Given a tensor $\mathcal{G}$ with the symmetric structure involving the first p indices, assume that $\mathcal{G}$ has nonzero sub-blocks with respect to $\mathbf{U} = (\mathbf{0}_l^\top \mathbf{I}_m)^\top$, that is, $\|\mathcal{G} \cdot \mathbf{U}^{\otimes p}\|_2 \neq 0$. For any positive number $K > 0$ and $p > 1$, there always exist a pair of vectors $\mathbf{h} \in \mathbb{R}^m$ and $\mathbf{x} \in \mathbb{R}^l$, such that $\|J_s(\mathbf{h}; \mathbf{x})\|_2 > K$, that is, the model (3) is unstable under Definition 2.

Theorem 3 implies that the TRU (3) is non-Lipschtiz-continuous when $p > 1$. In this case, the model would stay in the unstable regime according to Definition 2. It is known that the gradients of the loss function explode in unstable models [22]. Therefore, Theorem 3 partially reveals the insight into why most (high-degree) tensor models are practically difficult to train, even in the deep feedforward neural networks [4].

C. Discussion

The aforementioned results show a desperate picture: it seems difficult for the model to obtain the long memory effect meanwhile operating in a stable regime. To infer its causes, we conjecture that *the culprit is the discrete essence of the degree parameter* p. For instance, $p = 1$ (i.e., the linear form) provides the model superior stability with short memory; however, the $p = 2, 3, \ldots$ cases will result in unstable states and a potential long memory. It inspires us to seek the "edge" of such phase transition from the middle of integers (even less than one). As a consequence, we could benefit from both stability and long-term memory in practice. However, it is nontrivial to achieve the goal because the degree-p tensor

power is defined as the multiple folds of tensor products. To tackle the issue, in Section IV, we reformulate the weight tensor $\mathcal{G}$ in (3) using the well-known TD, which allows us to extend the feasible domain of p from the intrinsically discrete to a continuous domain. The extension gives the model the capability of spontaneously learning the optimal degree parameter with respect to the loss function.

IV. Degree-Learnable Approach

Below, we show how to extend the degree parameter to a continuous domain and empirically introduce a degree-learnable approach for the corresponding RNNs.

A. Model Description

Recall (3) yet adding the bias term for the empirical purpose

$$\mathbf{h}^{(t)} = \mathcal{G} \cdot \begin{pmatrix} \mathbf{x}^{(t)} \\ \mathbf{h}^{(t-1)} \end{pmatrix}^{\otimes p} + \mathbf{b} \tag{9}$$

where $\mathbf{b} \in \mathbb{R}^m$. Thus, the jth entry of the hidden state $\mathbf{h}^{(t)}$, written $\mathbf{h}^{(t)}[j]$, satisfies

$$\mathbf{h}^{(t)}[j] = \left\langle \mathcal{G}_j, \begin{pmatrix} \mathbf{x}^{(t)} \\ \mathbf{h}^{(t-1)} \end{pmatrix}^{\otimes p} \right\rangle + \mathbf{b}[j] \tag{10}$$

where $\mathcal{G}_j \in \mathbb{R}^{n^p}$ denotes the subblock of $\mathcal{G}$ by fixing the last index to equal j. Moreover, we know the tensor $\mathcal{G}_j$ is supersymmetric [23]. In this form, the parameter p is only defined in the range of nonnegative integers, that is, $p \in \mathbb{Z}^+$. Next, we apply symmetric TD [24] to decomposing the tensor $\mathcal{G}_j$ into factors, a.k.a., latent components [23]. As a result, (10) can be represented as

$$\mathbf{h}^{(t)}[j] = \sum_{r=1}^{R} \left\langle \mathbf{w}_{j,r}, \begin{pmatrix} \mathbf{x}^{(t)} \\ \mathbf{h}^{(t-1)} \end{pmatrix} \right\rangle^p + \mathbf{b}[j] \tag{11}$$

where $\mathbf{w}_{j,r} \in \mathbb{R}^n$ denotes the rth factor of $\mathcal{G}_j$ and $R > 0$ corresponds to the symmetric tensor rank. The work in [25] shows that the decomposition always exists for any symmetric tensor $\mathcal{G}_l$ if the rank R is sufficiently large, implying the equivalence between (10) and (11). Apart from the equivalence, we can also see that *the parameter p is converted into a vanilla exponent, which is defined explicitly on not $\mathbb{Z}^+$ but the real field $\mathbb{R}$*. It allows us to naturally extend the TRU to the "real" degree. However, given a noninteger p, note that the exponential term in (11) is defined only when the base (i.e., the inner product term) is positive. Therefore, we heuristically extract the sign of the base from the exponential function, that is,

$$\mathbf{h}^{(t)}[j] = \sum_{r=1}^{R} a_{j,r} \left| \left\langle \mathbf{w}_{j,r}, \begin{pmatrix} \mathbf{x}^{(t)} \\ \mathbf{h}^{(t-1)} \end{pmatrix} \right\rangle \right|^p + \mathbf{b}[j] \tag{12}$$

where $a_{j,r} := \operatorname{sgn}\left(\left\langle \mathbf{w}_{j,r}, \begin{pmatrix} \mathbf{x}^{(t)} \\ \mathbf{h}^{(t-1)} \end{pmatrix} \right\rangle \right)$. Given any $p \in \mathbb{R}$, define an element-wise nonlinear function $\phi_p(\cdot):\mathbb{R}^m \to \mathbb{R}^m$, of which each element $\phi_p(s)$ is given by

$$\phi_p(s) = \operatorname{sgn}(s) \cdot |s|^p \tag{13}$$

then, we finally have the extension of TRU, called fTRU, by concatenating (12) over all possible $j \in [m]$, that is,

$$\mathbf{h}^{(t)} = \sum_{r=1}^{R} \phi_p\left(\mathbf{W}_{hh,r} \mathbf{h}^{(t-1)} + \mathbf{W}_{hx,r} \mathbf{x}^{(t)} \right) + \mathbf{b} \tag{14}$$

where $\mathbf{W}_{hh,r} \in \mathbb{R}^{m \times m}$ and $\mathbf{W}_{hx,r} \in \mathbb{R}^{m \times l}$ are the weights, where the jth row of their concatenation corresponds to the vector $\mathbf{w}_{j,r}$ in (12).

It can be seen that the transition of the hidden states in (14) results in a *multibranch* neural network following the activation function ϕ_p, and the number of branches is determined by the maximum of the symmetric tensor rank of $\mathcal{G}_j, \forall j \in [l]$.

We can see that the degree-induced function ϕ_p can provide sufficient nonlinearity (when $p \neq 1$) to the learning model even if the standard "activations" are omitted. Moreover, recent work [13] shows that any saturated continuous activation functions like "tanh" and "sigmoid" lead to short memory of the model. Unlike them, the function ϕ_p is unbounded if $p \neq 0$, and its curvature is controllable in terms of the parameter p. Therefore, the model (14) can also be considered as an activation-learnable recurrent model, but which is expected to have the long-memory effect.

B. Application in RNNs

The model (14) can be directly employed to replace the original recurrent model in both vanilla RNN and LSTM. For the latter, we suggest a similar way as [3], that is,

$$\begin{aligned} \left[\mathbf{i}^{(t)}, \mathbf{g}^{(t)}, \mathbf{f}^{(t)}, \mathbf{o}^{(t)}\right] &= \sum_{r=1}^{R} \phi_p\left(\mathbf{W}_{hh,r} \mathbf{h}^{(t-1)} + \mathbf{W}_{hx,r} \mathbf{x}^{(t)} \right) + \mathbf{b} \\ \mathbf{c}^{(t)} &= \mathbf{c}^{(t-1)} \circ \mathbf{f}^{(t)}, \quad \mathbf{h}^{(t)} = \mathbf{c}^{(t)} \circ \mathbf{o}^{(t)} \end{aligned} \tag{15}$$

where $\circ$ denotes the Hadamard product. In addition, the trick by considering more historic states as [3] can be trivially applied to the model (14).

Besides the hidden states, we suggest two methods for learning the degree parameter p in the training phase. The most vanilla way is to consider p as a trainable variable. Since its feasible range becomes the whole real field $\mathbb{R}$, p can be efficiently trained by stochastic gradient descent (SGD) and its variants. The second way is to learn the parameter p by a subnetwork, that is,

$$p^{(t)} = \mathrm{MLP}\left(p^{(t-1)}, \mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}\right) \tag{16}$$

where $\mathrm{MLP}(\cdot)$ denotes a multilayer perceptron (MLP) and $p^{(t)}$ is the degree parameter at time step t. Unlike the first method, the subnetwork gives the model a variety of degree parameters at different time steps. From the empirical perspective, it is reasonable because the model would have an "attention-like" mechanism to selectively remember more important information from the data.

V. Experiments

In this section, we numerically show that the proposed fTRU model (14) can achieve superior performance on various time-series forecasting tasks in a stable manner. The

performance of different recurrent units in both single-cell and sequence-to-sequence (seq2seq) [26] structures are illustrated for comparison. After that, we discussed the influence of model structure on expressive ability, including tensor ranks, and used historical states. How degree p gets learned will be illustrated after that. Then we compare various tensor-based methods to demonstrate that the model stability can benefit from fTRU.

A. Performance Comparison

We compare fTRU with other recurrent units in both single-cell and seq2seq structures. To implement the first-stage simple experiment, we use single-cell recurrent network architecture and compare the models by employing them in an RNN framework, where the same datasets in [13] are used to demonstrate the effectiveness of the model, including "ARFIMA" [13], "Dow Jones Industrial Average (DJI)," "Traffic,[2]" and "tree-ring (Tree).[3]" ARFIMA is generated as a series of length 4001 using the model $(1-0.7B+0.4B^2)(1-B)^{0.4}Y_t = (1-0.2B)\varepsilon_t$ with obvious long memory effect. DJI contains the daily closing prices from 2000 to 2019 obtained from Yahoo Finance. Traffic contains the hourly Interstate 94 Westbound traffic volume for MN DoT ATR station 301, roughly obtained from the MN Department of Transportation. The tree contains 4351 tree ring measures of pine from India Garden, Nevada Gt Basin obtained from R package tsdl. Zhao et al.'s [13] work has shown that the four datasets have strong long-range dependence, that is, the long memory property. In the experiment, we perform one-step rolling forecasts on the test sets.

In the single-cell model, we use a two-layer MLP of the hidden dimension equal to 3 to calculate the degree parameter p and fix the tensor rank R in (14) to 1. Moreover, we also apply the trick mentioned in [3] by taking more historical states (one step more or none) into account, and the model is selected by choosing the one with the best performance on the validation set. The selection of tensor rank, historical states, and MLP layers will be discussed later. In the training phase, we apply the mean square error (MSE) as the loss function and employ the Adam algorithm with the learning rate equaling 0.01 for optimization, which stops after 1000 epochs. For comparison, we also report the performance of the various RNN models given in [13], which includes the vanilla RNN (RNN), two-lane RNN with the past 100 values as input (RNN2), recurrent weighted average network (RWA), MIxed hiSTory RNNs (MIST), memory-augmented RNN (MRNN), vanilla LSTM (LSTM), and memory-augmented LSTM (MLSTM). The recurrence architectures of selected models are illustrated in Table I. Persistence [27] as a classic statistic method is also included. All the experiments are run independently 50 times except persistence, as it will not be disturbed by random factors such as parameter initialization, where both the mean value and the standard deviation (std.) of the root-mean-square error (RMSE) on the test sets are illustrated.

TABLE I
RECURRENCE EQUATIONS OF THE SELECTED MODELS

	Update equation
RNN	$\mathbf{h}^{(t)} = \mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{W}\mathbf{x}^{(t)} + \mathbf{b}$
RWA	$\mathbf{h}^{(t)} = f(\frac{\sum_{i=1}^{t} z(\mathbf{x}^{(i)},\mathbf{h}^{(i-1)})\cdot e^{a(\mathbf{x}^{(i)},\mathbf{h}^{(i-1)})}}{\sum_{j=1}^{t} e^{a(\mathbf{x}^{(j)},\mathbf{h}^{(j-1)})}})$
MIST	$[\mathbf{a}^{(t)},\mathbf{r}^{(t)}] = \mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{W}\mathbf{x}^{(t)} + \mathbf{b}$ $\mathbf{h}^{(t)} = a(\mathbf{W}(\mathbf{r}^{(t)}) + \mathbf{W}\mathbf{x}^{(t)} \sum_{i=0}^{n_d-1} a_{ti} h_{t-2^i} + \mathbf{b})$
MRNN	$\mathbf{m}^{(t)} = a(\mathbf{W}\mathbf{m}^{(t-1)} + \mathbf{W}F(\mathbf{x}^{(t)}, d) + \mathbf{b})$
fTRU	$\mathbf{h}^{(t)} = \sum_{r=1}^{R} \phi_p(\mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{W}\mathbf{x}^{(t)}) + \mathbf{b}$

TABLE II
PERFORMANCE COMPARISON IN TERMS OF RMSE UNDER SINGLE-CELL STRUCTURE, WHERE THE AVERAGE AND STANDARD DEVIATION (IN BRACKETS) ARE REPORTED, AND THE BEST RESULTS ARE HIGHLIGHTED IN BOLD

	ARFIMA	DJI($\times$100)	Traffic	Tree
Persistence	1.914 (-)	0.328 (-)	378.98 (-)	0.345 (-)
RNN	1.1620 (0.1980)	0.2605 (0.0171)	336.44 (10.401)	0.2871 (0.0086)
RNN2	1.1630 (0.1820)	0.2521 (0.0112)	336.32 (10.182)	0.2855 (0.0077)
RWA	1.6840 (0.0050)	0.2689 (0.0095)	346.62 (1.410)	0.3048 (0.0001)
MIST	1.1390 (0.1832)	0.2604 (0.0154)	358.09 (16.270)	0.2883 (0.0091)
MRNN	1.0880 (0.1140)	**0.2487** (0.0105)	333.72 (10.157)	0.2818 (0.0053)
LSTM	1.1340 (0.1200)	0.2492 (0.0128)	337.60 (8.146)	0.2833 (0.0070)
MLSTM	1.1490 (0.1660)	0.2531 (0.0130)	337.83 (9.440)	0.2859 (0.0083)
Ours	**1.0691** (0.0245)	0.2672 (0.0526)	**329.22** (3.3713)	**0.2799** (0.0023)

The RMSE performance of those models is shown in Table II. We can see that our model (14) outperforms other models on "ARFIMA," "Traffic," and "Tree," and remains competitive on "DJI." Apart from the mean value, we can also see that our model demonstrates a smaller standard deviation than its counterparts, such as MRNN and MLSTM. Although RWA shows the smallest standard deviation in the experiment, its average performance is not comparable to ours. Moreover, Fig. 2 shows the average training time per epoch under various training sizes. Our model is twice as slow as the vanilla RNN yet ten times faster than MRNN, which has the closest performance to ours.

Next, we evaluate the effectiveness of our model in a deeper architecture, that is, sequence-to-sequence (seq2seq), on the forecasting task. We consider one synthetic dataset and two real-word datasets similar to [3], including "Genz" [3], "Traffic of Los Angeles County (TrafficLA),[4]" and "Solar.[5]" Genz functions are taken as the basic expression of higher-order functions without noise. We generate 1000 sequences at random initial points. TrafficLA provided by the California

[2]https://archive.ics.uci.edu/ml/datasets/Metro+Interstate+Traffic+Volume
[3]https://pkg.yangzhuoranyang.com/tsdl/
[4]http://pems.dot.ca.gov
[5]https://www.nrel.gov/grid/solar-power-data.html

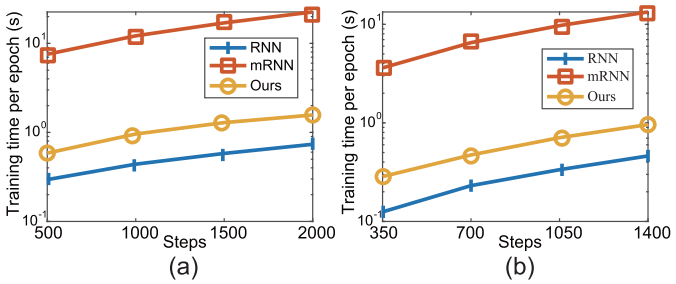


Fig. 2. Training time per epoch with various lengths of the training sets on (a) ARFIMA and (b) Traffic.

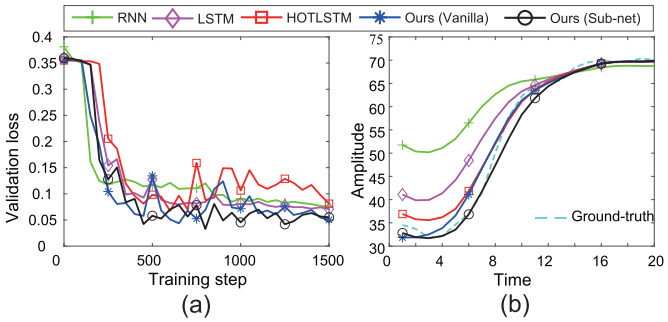


Fig. 3. (a) Validation dynamics on "Genz" during training and (b) visualization of the prediction on "TrafficLA."

TABLE III

BEST RMSE PERFORMANCE OF SEQ2SEQ MODELS ON THE TEST SETS, AND THE BEST RESULTS ARE HIGHLIGHTED IN BOLD

	Genz	TrafficLA	Solar
Persistence	0.0832	0.3010	0.2820
RNN	0.0172	0.0874	0.1273
MIRNN	0.0073	0.0860	0.1174
nnRNN	0.0148	0.0857	0.0977
LSTM	0.0085	0.0900	0.0947
HOTLSTM	0.0074	0.0851	0.0933
Ours (Vanilla)	0.0061	0.0853	0.0933
Ours (Sub-net)	**0.0058**	**0.0828**	**0.0926**

TABLE IV

COMPUTATIONAL COMPLEXITY RECORDS (ROUNDING TO 100) OF THE BEST-PERFORMED SEQ2SEQ MODELS

	Size	Time (s) Genz	TrafficLA	Solar
RNN	700	100	800	600
MIRNN	900	200	1600	800
nnRNN	1400	200	1800	1700
LSTM	3300	500	3700	2600
HOTLSTM	2600	400	2500	2000
Ours (Vanilla)	2400	300	2400	2000
Ours (Sub-net)	2900	300	2600	1900

Department of Transportation contains the speed readings collected by 325 speed sensors. Solar provided by NREL contains data points for 137 synthetic solar photovoltaic power plants in Alabama State. Unlike the one-step rolling forecast in the single-cell experiment, here we use partial observation to predict the rest of the time series, that is, relatively long-term prediction.

We follow a similar setup to the experiments given in [3]. In our model, we modify the LSTM cells in the seq2seq network as (15) and learn the degree parameters through both trainable variables (Vanilla) and subnetworks (Sub-net) as mentioned in Section IV-B. We also use MSE as the loss function and rms-prop as the optimizer to train the models. For comparison, seq2seq models with both RNN and LSTM are employed in the experiment. Multiplicative integrated RNN (MIRNN) proposed by [20] is included as an improved version of RNN. Various studies focus on enhancing the long-term memory of RNNs [28], [29], hence we introduce nonnormal RNN (nnRNN) from [29] into the experiment. We also compare the performance of our model with the high-order tensor LSTM (HOTLSTM) [3], which is the most related tensor method to ours. The hyperparameters of the above learning models are chosen by grid search. The values with the best performance on the validation set are selected. The search range and more details are described in Supplement of Section 3.C. The classic persistence method is also included for comparison.

Table III shows the best RMSE performance of those models on the three datasets. In our models, "vanilla" refers to direct learning with the degree parameters as trainable variables, and "Sub-net" means the degree parameters are learned by subnetworks. We can see that our models outperform not only the conventional RNN, LSTM, MIRNN, and nnRNN, but also the more advanced tensor-based model HOTLSTM. We also show the dynamics of validation loss on "Genz" and the prediction visualization on "TrafficLA" in Fig. 3. We can see that our model (Sub-nets, the black line) consistently obtains the superior validation loss while growing the training steps, and the prediction results visually demonstrate the effectiveness of our model.

Computational complexity is an essential aspect of performance to evaluate the learning efficiency of deep-learning models. We record the computational complexity including model sizes and training times of each seq2seq learning method with details in Table IV. The training time is calculated by the average training seconds at a Tesla V100-PCIE-32 GB GPU in all runs. It is worth mentioning that to avoid wasting computational resources and overfitting, we use both the maximum training epoch and the early termination mechanism. This means that the model automatically stops training and starts evaluation once it reaches the upper limit of the training epoch or the loss does not decrease within the given steps. Hence, the training time of different models is not exactly proportional to the model size, as their training process can be accomplished with different epochs. Based on the complexity statistics, the performance of our proposed model is also comparable.

B. Model Architecture

The selection of fTRU's inner structure is discussed here, including tensor rank, used historical states, and MLP layers. Table V shows the results of our single-cell model on "ARFIMA" with various tensor rank R. We can see a

TABLE V
PERFORMANCE OF OUR SINGLE-CELL MODEL ON "ARFIMA" UNDER THE VARIOUS RANKS

	RMSE		Training loss		
	mean	std.	mean	std.	kurtosis
$R = 5$	1.0851	0.0270	0.0078	0.0003	2.0620
$R = 10$	1.0963	0.0261	0.0079	0.0003	2.6152
$R = 30$	1.1057	0.0315	0.0081	0.0004	3.2933
$R = 50$	1.1103	0.0316	0.0081	0.0004	3.6301

TABLE VI
TEST RMSE OF THE PROPOSED SINGLE-CELL MODEL ON DATASETS "ARFIMA" AND "TREE"

	ARFIMA		Tree	
	mean	std.	mean	std.
$D_h = 1$	1.0828	0.0353	**0.2799**	0.0023
$D_h = 2$	**1.0691**	0.0245	0.2803	0.0021
$D_h = 3$	1.0741	0.0388	0.2805	0.0022
$D_h = 5$	1.0743	0.0348	0.2803	0.0021
$D_h = 10$	1.0835	0.0357	0.2814	0.0018

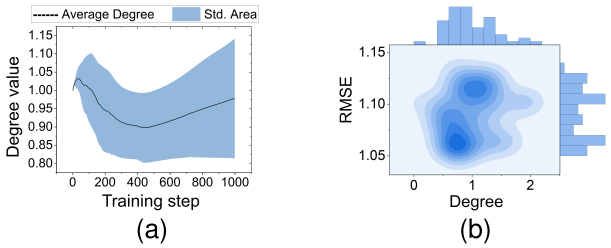


Fig. 4. (a) Trend of degree p value during the training on "ARFIMA" in 50 runs. (b) Distribution of degree p and corresponding test RMSE after training.

TABLE VII
STATISTICS OF THE RMSE AND LEARNED p OF VANILLA SINGLE-CELL FTRU ON DATASETS, EXCEPT THREE UNSTABLE SAMPLES ON "DJI"

	Learned p		RMSE	
	mean	std.	mean	std.
ARFIMA	0.9783	0.4050	1.0898	0.0240
DJI*	1.2630	0.8138	0.0029	0.0001
Traffic	1.2599	0.4245	328.91	3.0156
Tree	1.3192	0.5736	0.2816	0.0017

counter-intuitive phenomenon that a larger tensor rank would neither improve (even degrade) the prediction accuracy nor decrease the training loss. Similar results have also been reported in tensor literature [9]. It implies that the rank would not be a key factor to determine the approximation capacity of the model.[6] Additionally, it is worth noting that, while increasing the rank, although all the averages of loss are almost unchanged, the kurtosis (a concentration measurement of the distribution) significantly increases. We infer that the model with large ranks generally has a more flat landscape in the training phase, that is, the loss is less nonconvex, such that gradient-based optimizers less influence the well-trained loss values with different initialization. This result is empirically consistent with a recent study on the landscape of multibranch feedforward neural networks [30].

Table VI shows the test RMSE of our single-cell model on "ARFIMA" and "Tree" armed with the various numbers of historical states as [3], where $D_h = 1$ denotes only the current hidden state is used. It can be seen that the additional states can improve the prediction accuracy. We also infer that more historical information on the hidden states can enhance the short-term prediction capability of the model. On the other hand, the performance could be worse if too much "history" is used. We conclude that in this case, the short-term contribution can *dominate* the prediction performance, such that the model cannot exploit the long memory effect well.

C. Degree Learning Process

The learning approach of the degree p and related results are discussed in this part. In our method, the degree p is initialized with a suitable positive boundary constant, such as 1, and then optimized over the training set. The feasible range of degree p is not constrained. We use the current state as $D_h = 1$ here and train the single-cell model for 1000 epochs with 50 runs on ARFIMA for the illustration. We can see in Fig. 4(a) that after different initialization, the degree during training has a similar central tendency in most cases. Moreover, the numerical values of the degree and test error also illustrate correlation and clustering as Fig. 4(b) shown. The statistical results of RMSE and p on various datasets are shown in Table VII, in which we have found that the optimized degree values can be different on the various tasks, yet mostly close to 1. This support the necessity of learning tensor degree in the fractional domain. Three samples of DJI are not included due to the unstable training process, which will be discussed in the next section.

In the single-cell subnet fTRU model, we also find similarity exists in the distribution of degree p on time-series data, where the degree value is independent at each time step. Generally, there exist two types of distribution in ARFIMA, with the results from 19 and 30 times out of 50 total runs. In Fig. 5(a), they are visualized in the average value, where we can find significant similarities in time steps. It also shows the connection between the distribution of degree and hidden states, as the vertical trend of hidden value is generally opposite to the degree. In a few cases, the degree value and the corresponding hidden state oscillate on time steps, fluctuating between two values, as shown in Fig. 5(b). The test RMSE values of those three types result in 1.09($\pm$0.02), 1.07($\pm$0.02), and 1.11.

D. Model Stability

We achieve the stability analysis of various tensor-based recurrent models on ARFIMA for comparison. For comparison, we take origin tensor-power RNN (origin TRNN), tensor-power RNN with tensor canonical polyadic decomposition (CP TRNN), and the most advanced tensor-train RNN (T-T RNN). They all have rank = 2 in the experiment. The initialization methods applied in this section are different from the ones that perform best in Table II, as we take those that can

[6]Note that it is a different issue from the discussion in [11].

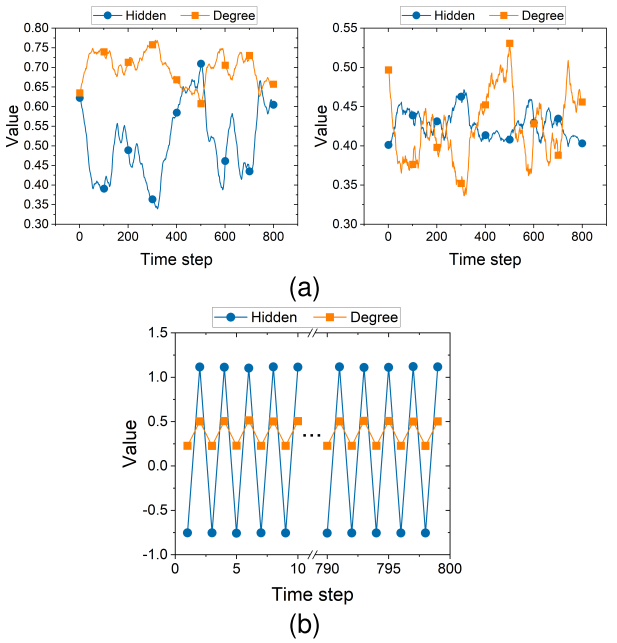


Fig. 5. Trends of degree p and hidden value on time steps in the test sets. (a) Trends of two types of centralization, including the mean value of degree p and the corresponding hidden states on each time step smoothed by the Savitzky–Golay filter [31]. (b) Specific sample with the oscillating performance of both values.

TABLE VIII
STATISTICS OF STABILITY OF THE DIFFERENT TENSOR MODELS UNDER DIFFERENT DEGREES ON "ARFIMA," WITH TAKING GRADIENT EXPLOSION AS THE UNSTABLE BEHAVIOR

		Stable percentage in 50 runs			
Degree	Dynamic	2	4	6	8
Origin TRNN	(-)	0.34	0.04	0.02	0.00
CP TRNN	(-)	0.64	0.70	0.70	0.72
T-T RNN	(-)	0.40	0.18	0.10	0.00
Ours (Vanilla)	0.72	(-)	(-)	(-)	(-)
Ours (Sub-net)	1.00	(-)	(-)	(-)	(-)

better demonstrate the stability differences among the various fixed degrees. The degree of fTRU is "dynamic" because the degree is not fixed but learnable in our approach. If the gradient explosion happens in the training process, we take this run as unstable. Probabilities of tensor-based models remaining stable during training are shown in Table VIII. It is found that increasing the fixed degree has significant effects on the origin tensor-power and tensor-train models and results in unstable training, while fTRU shows better stability, especially the subnet model. The canonical polyadic decomposition model can maintain its stability during the increasing of tensor degree, while still performing more unstable compared with fTRU. It can be concluded that the proposed fractional method performs with higher stability in the comparison of various advanced tensor recurrent models, which is consistent with the proposed theorem.

VI. CONCLUSION

In this article, we propose an fTRU to improve the expressive ability of RNNs in a stable manner with learnable fractional tensor operations. Our theoretical results show that the degree parameter plays a crucial role in both the memory mechanism and dynamic behaviors of tensor models. Our experimental results show fTRU's contributions to the expressive ability and stability of tensor recurrent models. The superiority of our presented method is demonstrated in both synthetic and real-world forecasting tasks. In the future, we plan to extend the proposed fractional method to more applications, such as classification and translation systems.

REFERENCES

[1] I. Sutskever, J. Martens, and G. E. Hinton, "Generating text with recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 1017–1024.
[2] I. Schlag and J. Schmidhuber, "Learning to reason with third order tensor products," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 9981–9993.
[3] R. Yu, S. Zheng, A. Anandkumar, and Y. Yue, "Long-term forecasting using higher order tensor RNNs," 2017, *arXiv:1711.00073*.
[4] J. Su, W. Byeon, J. Kossaifi, F. Huang, J. Kautz, and A. Anandkumar, "Convolutional tensor-train LSTM for spatio-temporal learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 13714–13726.
[5] J. Ye et al., "Learning compact recurrent neural networks with block-term tensor decomposition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9378–9387.
[6] Y. C. Lee et al., "Machine learning using a higher order correlation network," *Phys. D, Nonlinear Phenomena*, vol. 22, nos. 1–3, pp. 276–306, Oct. 1986.
[7] E. Bas, E. Egrioglu, and E. Kolemen, "Training simple recurrent deep artificial neural network for forecasting using particle swarm optimization," *Granular Comput.*, vol. 7, no. 2, pp. 411–420, Apr. 2022.
[8] T. Cansu, E. Kolemen, Ö. Karahasan, E. Bas, and E. Egrioglu, "A new training algorithm for long short-term memory artificial neural network based on particle swarm optimization," *Granular Comput.*, vol. 8, no. 6, pp. 1645–1658, Nov. 2023.
[9] B. Li, C. Li, F. Duan, N. Zheng, and Q. Zhao, "TPFN: Applying outer product along time to multimodal sentiment analysis fusion on incomplete data," in *Proc. Eur. Conf. Comput. Vis.*, vol. 16, pp. 431–447, 2020.
[10] A.-H. Phan, A. Cichocki, I. Oseledets, G. G. Calvi, S. Ahmadi-Asl, and D. P. Mandic, "Tensor networks for latent variable analysis: Higher order canonical polyadic decomposition," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 6, pp. 2174–2188, Jun. 2020.
[11] V. Khrulkov, A. Novikov, and I. Oseledets, "Expressive power of recurrent neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2018.
[12] T. Trinh, A. Dai, T. Luong, and Q. Le, "Learning longer-term dependencies in RNNs with auxiliary losses," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4965–4974.
[13] J. Zhao et al., "Do RNN and LSTM have long memory?" in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 11365–11375.
[14] J. Miller and M. Hardt, "Stable recurrent models," in *Proc. Int. Conf. Learn. Represent.*, 2018.
[15] P. Royston and D. G. Altman, "Regression using fractional polynomials of continuous covariates: Parsimonious parametric modelling," *Appl. Statist.*, vol. 43, no. 3, p. 429, 1994.
[16] Y.-F. Pu, Z. Yi, and J.-L. Zhou, "Fractional Hopfield neural networks: Fractional dynamic associative recurrent neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2319–2333, Oct. 2017.
[17] H. Qiu, C. Li, Y. Weng, Z. Sun, X. He, and Q. Zhao, "On the memory mechanism of tensor-power recurrent models," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2021, pp. 3682–3690.
[18] C. Li, Z. Sun, and Q. Zhao, "High-order learning model via fractional tensor network decomposition," in *Proc. 1st Workshop Quantum Tensor Netw. Mach. Learn., 34th Conf. Neural Inf. Process. Syst.*, 2020.
[19] A. Greaves-Tunnell and Z. Harchaoui, "A statistical investigation of long memory in language and music," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2394–2403.
[20] Y. Wu, S. Zhang, Y. Zhang, Y. Bengio, and R. R. Salakhutdinov, "On multiplicative integration with recurrent neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2856–2864.
[21] R. Tomioka and T. Suzuki, "Spectral norm of random tensors," 2014, *arXiv:1407.1870*.

[22] M. Hardt, T. Ma, and B. Recht, "Gradient descent learns linear dynamical systems," *J. Mach. Learn. Res.*, vol. 19, no. 1, pp. 1025–1068, 2018.
[23] A. Cichocki, R. Zdunek, and S.-I. Amari, "Nonnegative matrix and tensor factorization," *IEEE Signal Process. Mag.*, vol. 25, no. 1, pp. 142–145, Mar. 2008.
[24] J. Brachat, P. Comon, B. Mourrain, and E. Tsigaridas, "Symmetric tensor decomposition," *Linear Algebra Appl.*, vol. 433, nos. 11–12, pp. 1851–1872, Dec. 2010.
[25] P. Comon, G. Golub, L.-H. Lim, and B. Mourrain, "Symmetric tensors and symmetric tensor rank," *SIAM J. Matrix Anal. Appl.*, vol. 30, no. 3, pp. 1254–1279, Jan. 2008.
[26] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.
[27] G. M. Caporale and N. Pittis, "Persistence in macroeconomic time series: Is it a model invariant property?" London Bus. School Centre Econ. Forecasting, London, U.K., Tech. Rep. 02-96, 1996.
[28] E. Vorontsov, C. Trabelsi, S. Kadoury, and C. Pal, "On orthogonality and learning recurrent networks with long term dependencies," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 3570–3578.
[29] G. Kerg et al., "Non-normal recurrent neural network (NNRNN): Learning long time dependencies while improving expressivity with transient dynamics," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 13613–13623.
[30] H. Zhang, J. Shao, and R. Salakhutdinov, "Deep neural networks with multi-branch architectures are intrinsically less non-convex," in *Proc. 22nd Int. Conf. Artif. Intell. Statist.*, 2019, pp. 1099–1109.
[31] A. Savitzky and M. J. E. Golay, "Smoothing and differentiation of data by simplified least squares procedures," *Anal. Chem.*, vol. 36, no. 8, pp. 1627–1639, Jul. 1964.

Hejia Qiu received the bachelor's degree from the University of Nottingham Ningbo China, Ningbo, China, in 2019, where he is currently pursuing the Ph.D. degree with the School of Computer Science.

His research interests include machine learning and healthcare applications.

Chao Li received the bachelor's and Ph.D. degrees from Harbin Engineering University (HEU), Harbin, China, in 2006 and 2017, respectively.

He has been an Indefinite-Term Research Scientist at RIKEN, Tokyo, Japan, since 2021, working at the Center for Advanced Intelligence Project (AIP) together with Qibin Zhao. Before that, he was a Post-Doctoral Researcher with RIKEN-AIP from 2018 to 2020. During the Ph.D., he worked as an International Program Associate working with Prof. Andrzej Cichocki at RIKEN Brain Science Institute, Wako, Japan, from 2013 to 2015. In five years, he has published more than 50 academic papers in conferences and journals such as International Conference on Machine Learning (ICML), Conference on Neural Information Processing Systems (NeurIPS), International Conference on Artificial Intelligence and Statistics (AISTATS), Conference on Computer Vision and Pattern Recognition (CVPR), European Conference on Computer Vision (ECCV), AAAI Conference on Artificial Intelligence (AAAI), IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (T-PAMI), IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS (T-NNLS), and IEEE TRANSACTIONS ON IMAGE PROCESSING (T-IP). His research interests include tensor networks and machine learning.

Dr. Li received the RIKEN Ohbu Research Incentive Award in 2020. He regularly serves as a Senior Reviewer for ICML, NeurIPS, International Conference on Learning Representations (ICLR), AAAI, International Joint Conference on Artificial Intelligence (IJCAI), and so on.

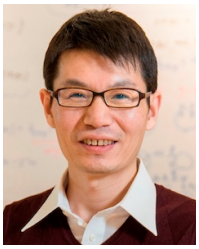
Ying Weng received the Ph.D. degree from the Chinese Academy of Sciences, Beijing, China, in 2005.

Before joining as a Lecturer (Assistant Professor) at the School of Computer Science, Bangor University, Bangor, U.K., in 2011, she was a Researcher at British Broadcasting Corporation Research and Development, London, U.K., and a Post-Doctoral Researcher at the Department of Electrical and Electronic Engineering, Imperial College London, London. She is currently an Associate Professor and the Computer Science Course Director at the School of Computer Science, University of Nottingham Ningbo China, Ningbo, China. She has published over 60 refereed research articles. Her research interests include machine learning and healthcare applications, computer vision, big data on image/video processing, the Internet of Things, multimedia security and forensics, and QoS in wireless networks.

Dr. Weng received the Best Paper Award from the 13th International Conference on Automation and Computing in 2007, the Best Poster Competition Award from the world's largest Alzheimer's Association International Conference (AAIC) in 2023, and the First Prize Award from Brain Tumor Segmentation (BraTS) Image Synthesis (Inpainting) in the world largest Medical Image Computing and Computer Assisted Intervention Society (MICCAI) Cluster of Challenges 2023. She served as the Chair for the 46th IEEE International Conference on Communications Wireless Communications Symposium: Resource Allocation, Multiuser MIMO in 2011.

Zhun Sun received the Ph.D. degree in information science from Tohoku University, Sendai, China, in 2018.

She has been working with the RIKEN Center for Advanced Intelligence Project, Tokyo, Japan, as a Post-Doctoral Researcher until 2020. She is currently working with the Graduate School of Information Sciences, Tohoku University. Her main research topics include computer vision with deep learning, generative models, computational graphics, and other deep neural network-related tasks.



Qibin Zhao (Senior Member, IEEE) received the Ph.D. degree in computer science from Shanghai Jiao Tong University, Shanghai, China, in 2009.

He was a Research Scientist at RIKEN Brain Science Institute, Wako, Japan, from 2009 to 2017. He joined RIKEN Center for Advanced Intelligence Project, Tokyo, Japan, as a Unit Leader, from 2017 to 2019, and is currently a Team Leader for Tensor Learning Team. He is also a Visiting Professor at the Tokyo University of Agriculture and Technology, Tokyo. He has published more than 150 scientific articles and coauthored two monographs on tensor networks. His research interests include machine learning, tensor factorization and tensor networks, and brain signal processing.

Dr. Zhao serves as an Area Chair for top-tier ML conferences of Conference on Neural Information Processing Systems (NeurIPS), International Conference on Machine Learning (ICML), International Conference on Learning Representations (ICLR), International Conference on Artificial Intelligence and Statistics (AISTATS), and Asian Conference on Machine Learning (ACML), and an Action Editor for *Neural Networks*, *Transactions on Machine Learning Research*.