



## OPEN ACCESS

## EDITED BY

Guitao Cao,  
East China Normal University, China

## REVIEWED BY

Yancheng Ji,  
Nantong University, China  
Yong Cai,  
Shanghai Astronomical Observatory  
(CAS), China

## \*CORRESPONDENCE

Chunhua Qian  
✉ chqian2003@126.com  
Rui Wang  
✉ rwang@shu.edu.cn

RECEIVED 24 November 2022

ACCEPTED 01 December 2022

PUBLISHED 22 December 2022

## CITATION

Li Y, Wang Y, Wang Y, Qian C and  
Wang R (2022) Geometric algebra  
based recurrent neural network for  
multi-dimensional time-series  
prediction.  
*Front. Comput. Neurosci.* 16:1078150.  
doi: 10.3389/fncom.2022.1078150

## COPYRIGHT

© 2022 Li, Wang, Wang, Qian and  
Wang. This is an open-access article  
distributed under the terms of the  
[Creative Commons Attribution License  
\(CC BY\)](https://creativecommons.org/licenses/by/4.0/). The use, distribution or  
reproduction in other forums is  
permitted, provided the original  
author(s) and the copyright owner(s)  
are credited and that the original  
publication in this journal is cited, in  
accordance with accepted academic  
practice. No use, distribution or  
reproduction is permitted which does  
not comply with these terms.

# Geometric algebra based recurrent neural network for multi-dimensional time-series prediction

Yanping Li<sup>1,2</sup>, Yi Wang<sup>1</sup>, Yue Wang<sup>1</sup>, Chunhua Qian<sup>3\*</sup> and Rui Wang<sup>1\*</sup>

<sup>1</sup>School of Communication and Information Engineering, Shanghai University, Shanghai, China, <sup>2</sup>Office of Academic Affairs, Shanghai University, Shanghai, China, <sup>3</sup>Department of Endocrinology and Metabolism, Shanghai Tenth People's Hospital, School of Medicine, Tongji University, Shanghai, China

Recent RNN models deal with various dimensions of MTS as independent channels, which may lead to the loss of dependencies between different dimensions or the loss of associated information between each dimension and the global. To process MTS in a holistic way without losing the inter-relationship among dimensions, this paper proposes a novel Long-and Short-term Time-series network based on geometric algebra (GA), dubbed GA-LSTNet. Specifically, taking advantage of GA, multi-dimensional data at each time point of MTS is represented as GA multi-vectors to capture the inherent structures and preserve the correlation of those dimensions. In particular, traditional real-valued RNN, real-valued LSTM, and the back-propagation through time are extended to the GA domain. We evaluate the performance of the proposed GA-LSTNet model in prediction tasks on four well-known MTS datasets, and compared the prediction performance with other six methods. The experimental results indicate that our GA-LSTNet model outperforms traditional real-valued LSTNet with higher prediction accuracy, providing a more accurate solution for the existing shortcomings of MTS prediction models.

## KEYWORDS

geometric algebra, recurrent neural network, long-and short-term time-series network, prediction, multi-dimensional time-series

## 1. Introduction

Multi-dimensional time-series (MTS) are ubiquitous in our daily lives, including stock market prices, traffic flow on highways, output from solar power plants, temperatures in different cities, and so on. Prediction of these time-series can serve as the basis for many practical applications. However, there are usually complex dynamic interdependencies among the variables of these data (Faloutsos et al., 2018), and how to capture and utilize this information for efficient and reliable prediction is a long-standing research hotspot.

Traditional methods for MTS prediction include linear support vector regression (Cao and Tay, 2003), autoregressive integrated moving average models (Gurand and Whittle, 1951), vector autoregression (VAR) (Han et al., 2015), and so on. The most common approach is to treat observations at a point in time as vectors and model dynamic time using VARs and linear dynamical systems. Essentially, these models rely on AR coefficient matrices or dynamical matrices to capture the correlation structure between different time-series. Chen et al. (2021) extended the VAR model for third-order tensor time-series by introducing two AR coefficient matrices to characterize the correlation structure. Mohammad et al. (2019) also achieved excellent results in time-series data prediction using a stochastic model approach. However, the large number of parameters in the coefficient matrix and the high computational cost make these model parameters very difficult to estimate and prone to overfitting. At the same time, the performance of these traditional models in time-series with mixed long- and short-term modes is always insufficient, and cannot accurately capture the complex nonlinear relationship between sequence data.

Due to the excellent performance of deep learning in applications such as image recognition and machine translation, its potential in the field of MTS prediction has also attracted a lot of attention. Recent studies have indicated that modern deep learning techniques not only achieve state-of-the-art prediction performance but also systematically reduce the complexity of the prediction process significantly, thus improving maintainability (Hochreiter and Schmidhuber, 1997).

Recurrent neural network (RNN) and long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) are the earliest neural networks to deal with time-series. Subsequently, the emergence of gated recurrent units (Cho et al., 2014) reduces the number of network parameters, reducing the risk of overfitting compared to LSTMs. At the same time, there are also many works showing that specific convolutional neural network structures can also achieve good results, such as convolution-based gated linear units (Dauphin et al., 2017) and temporal convolutional networks (Bai et al., 2018). Due to the complex structural information among the dimensions of multi-dimensional time-series, it is difficult for a single network to achieve a good processing effect. Therefore, the emergence of hybrid deep networks has brought prediction accuracy to a new level. LSTNet (Ai et al., 2017) combines CNN, LSTM, attention mechanism, and AR autoregressive process to extract short-term local dependence patterns among variables and long-term dependence patterns of time-series. DeepState (Syama et al., 2018) combines state-space models with deep recurrent neural networks to learn the parameters of the whole network by maximum log-likelihood function. DeepGLO (Sen et al., 2019) is a hybrid model that includes a global matrix decomposition model normalized by a temporal convolutional network and a temporal network that captures the local properties of each time-series and associated covariates.

In addition, graph convolutional neural networks (GCNs) have also been shown to capture the correlation between partial time-series. Spatio-temporal graph convolutional neural network (ST-GCN) (Yu et al., 2018) is a deep learning framework for traffic prediction, which fully exploits the graph structure of road networks by integrating graph convolutions and gated linear units for faster training. Li et al. (2017) directly stacked graph convolution and temporal modules to capture spatial and temporal dependencies in traffic data streams, but the network requires predefined relational topology. Graph wavenet (Wu et al., 2019) combines graph convolution layers, adaptive adjacency matrices, and expanded stochastic convolution to capture spatio-temporal dependencies. However, most of them either ignore the correlation between data or require reliance on the graph as a priori. In addition, the Fourier transform has shown its advantages in previous work, especially the joint Fourier transform (Grassi et al., 2018; Isufi et al., 2019; Loukas and Perraudin, 2019), which enables prediction tasks in the fields of weather information, traffic data, and seismic waveforms. The discrete Fourier transform can also be used for time-series analysis. For example, state-frequency memory networks (Zhang et al., 2017) combine the advantages of the discrete Fourier transform and LSTM for stock price prediction together. Nevertheless, none of the existing solutions jointly captures temporal patterns and multivariate correlations in the spectral (Parcollet et al., 2019).

Accurate prediction based on historical time-series data is challenging because it requires joint modeling of the temporal patterns of the data and correlations among the data. How to capture and exploit the dynamic correlations among multiple variables is a huge research challenge for MTS prediction. At present, there are more studies based on multi-dimensional time-series prediction, however, they are all based on the real number, and there is no more advanced theoretical breakthrough. Moreover, in the process of multi-dimensional time-series processing and final fusion, real-valued networks have the problem of information loss inevitably. It is worth noting that Parcollet et al. (2019) constructed new quaternion recurrent neural networks and quaternion long short-term memory networks with quaternions, placing the three feature values of speech signals on each of the three imaginary parts of the quaternion, and exploiting the potential structural dependence within the quaternion to obtain better performance than real RNNs and LSTMs in practical automatic speech recognition applications. However, for signals with more features, such as MTS that usually have dozens or even hundreds of features, quaternion recurrent neural networks and quaternion long short-term memory networks are powerless.

Geometric algebra (GA) has opened up new directions for the study and application of MTS. Through the potential structural dependence within the multi-vector, the multi-dimensional features are combined into a single entity input to the network for processing, capturing the internal

relationships between the sequence features, allowing the structural information inherent in the multi-dimensional features to be well preserved.

Therefore, this paper proposes new geometric algebra based recurrent neural network (GA-RNN) and geometric algebra based long-and short-term time-series network (GA-LSTM) and constructs a new geometric algebra based long-and short-term time-series network, (GA-LSTNet). Firstly, the multi-dimensional time-series is represented as a GA multi-vector, preserving the correlation states of its channels. Secondly, each layer of the network and the training algorithm are extended to the GA space, and the corresponding processing algorithm is provided for the input GA multi-vector to ensure the retention of multi-channel information in signal processing. Essentially, each feature of the multi-dimensional signal is mapped to each component of the GA multi-vector separately, and then the overall computation based on the GA multi-vector is performed to maximize the retention of the potential features of the multi-dimensional signal.

The rest of this paper is organized as follows. Section 2 introduces the basics of GA and neural networks based on GA. Section 3 describes the proposed GA-RNN and GA-LSTM. Comparison experimental results between GA-LSTNet and real-valued methods are provided in Section 4, followed by concluding remarks drawn in Section 5.

## 2. Preliminary

GA was first described by William K. Clifford, also called Clifford Algebra. For multi-dimensional signals, GA is not only an effective framework to handle the representation and computation issues but also a useful tool for widespread use in mathematics and physics (Hestenes, 1986; Rafal, 2004; López-González et al., 2016).

Mathematically, suppose  $\mathbb{G}_n$  denotes a  $2^n$  dimensional vector space, and there exist a set of orthogonal bases  $\{e_1, e_2, \dots, e_n\}$ . The power set  $\gamma = \{1, \dots, n\}$  can turn the basis into an ordered one with the index set  $\Gamma$ .

$$\Gamma := \{(a_1, \dots, a_r) \in \gamma, 1 \leq a_1 \dots a_r \leq n\} \tag{1}$$

Then, the basis of  $\mathbb{G}_n$  is denoted by

$$\{e_I := e_{a_1} \dots e_{a_r} \mid I \in \Gamma\} \tag{2}$$

For example, the basis in the  $2^3$  vector space can be described as

$$\{1, e_1, e_2, e_3, e_{12}, e_{13}, e_{23}, e_{123}\}. \tag{3}$$

For convenience, in the rest of the paper,  $e_1 \dots e_r$  will be denoted by  $e_{1\dots r}$ . In general, the multiplication in GA will follow

the following rules

$$\begin{cases} e_i^2 = 1, & i = 1, \dots, p \\ e_i^2 = -1, & i = p + 1, \dots, p + q \end{cases} \tag{4}$$

and  $\mathbb{G}_n$  can also be denoted as  $\mathbb{G}_{p,q}$ , with  $2^n = p + q$ . An arbitrary element of the GA is given by

$$x = \sum_{t=0}^n \langle x \rangle_t = \sum_{I \in \Gamma} [x]_I e_I \tag{5}$$

Where  $[x]_A \in \mathbb{R}$ , represents the value of each component of the multi-vector. For example, an element in the  $2^3$  vector space can be represented as

$$\begin{aligned} x &= \langle x \rangle_0 + \langle x \rangle_1 + \langle x \rangle_2 + \langle x \rangle_3 \\ &= x_0 + x_1 e_1 + x_2 e_2 + x_3 e_3 \\ &\quad + x_{12} e_{12} + x_{13} e_{13} + x_{23} e_{23} + x_{123} e_{123} \end{aligned} \tag{6}$$

The addition in GA can be defined as

$$x + y = \sum_{I \in \Gamma} ([x]_I + [y]_I) e_I \tag{7}$$

The geometric product in GA can be written in the following form

$$x \otimes_{p,q} y = x \cdot y + x \wedge y \tag{8}$$

Where  $x \cdot y$  and  $x \wedge y$  represent the inner and outer products in GA, respectively.

The geometric product between two multi-vectors can also be converted into matrix operations. Assuming that there is a multi-vector, the multi-vector can be expressed as

$$\begin{aligned} x &= [[x]_0, [x]_1, [x]_2, \dots, [x]_I, \dots] \cdot [1, e_1, e_2, \dots, e_I, \dots]^T \\ &= F_x \cdot N_x \end{aligned} \tag{9}$$

Where  $F_x \in \mathbb{R}^{1 \times 2^n}$  is the coefficient matrix of multi-vector  $x$  and  $N_x$  is the corresponding orthogonal basis matrix. According to the calculation rules between different  $e_I$ ,  $R(x)$  can be defined as its real representation matrix (Roy et al., 2020). Then,

$$x \otimes_{p,q} y = [R(y) \cdot (F_x)^T] \cdot N_{x(y)} \tag{10}$$

The inversion of multi-vector is denoted by

$$\tilde{x} = \sum_{t=0}^n (-1)^{\frac{t(t-1)}{2}} \langle x \rangle_t \tag{11}$$

The conjugation of multi-vector is denoted by

$$x^* = \sum_{t=0}^n (-1)^{\frac{t(t+1)}{2}} \langle x \rangle_t \tag{12}$$

For any two multi-vectors  $x, y \in \mathbb{G}_n$ , dot product is defined by

$$x \odot y = \sum_{i \in \Gamma} [x]_I [y]_I e_I \tag{13}$$

In addition, similar to quaternion, the basic element of GA also has the concept of module. For any multi-vector, its module is defined by

$$\|x\| = \sqrt{\sum_{I \in \Gamma} ([x]_I)^2} \tag{14}$$

### 3. Methods

GA provides a new direction for the research and application of MTS. Through the latent structural dependencies within the multi-vector, the multi-dimensional features are combined into a single entity as input for network processing, and the internal relationship between the sequence features is captured, so that the inherent structural information in the multi-dimensional features is well preserved.

In this section, we extend RNN and LSTM from the real-value domain to the GA domain. In our proposed networks, inputs, outputs and weights are represented by GA multi-vectors. The operations in each layer and the training algorithm will be introduced in the following.

#### 3.1. Geometric algebra based recurrent network layer

The learning process of the geometric algebra based RNN layer (GA-RNN) is similar to that of the real-valued RNN, the difference is that the input and network parameters have become multi-vectors, as shown in Figure 1. The multi-dimensional features at each time point are converted into multi-vectors as input for GA-RNN. The weights of the input features are also multi-vectors.

Suppose that the dimension of input vector  $x_t$  at time  $t$  is  $N$ , the number of neurons in the hidden layer is  $H$  and the number of neurons in output layer is  $K$ . In addition,  $\sigma$  and  $\alpha$  represent the sigmoid and tanh activation functions, respectively. Then the forward propagation process of the GA-RNN basic unit is denoted by

$$\begin{aligned} a^t &= \sigma \left( U \otimes_{p,q} x^t + W \otimes_{p,q} a^{t-1} + \theta^a \right) \\ b^t &= \alpha \left( V \otimes_{p,q} a^t + \theta^b \right) \\ a^t &= \left[ a_1^t \dots a_h^t \dots a_H^t \right]^T \\ a^t &= \left[ b_1^t \dots b_k^t \dots b_K^t \right]^T \end{aligned} \tag{15}$$

Where  $x^t$  and  $a^t$  are multi-vector formed from the original input real-valued data and hidden state, respectively.  $U, W$ , and  $V$  are the weight matrices of the input, hidden state and output, respectively.  $\theta^a$  and  $\theta^b$  are the bias terms of the hidden state and output layer, respectively.  $b^t$  is the output target.

In addition, for a multi-vector  $x$ , assuming that  $f$  is any standard activation function, then

$$f(x) = \sum_{I \in \Gamma} f([x]_I) e_I \tag{16}$$

The activation function of multi-vector output is essentially the activation function operation in the real domain for each component of the multi-vector.

#### 3.2. Geometric algebra based back-propagation through time

The principle of back-propagation algorithm through time of GA-RNN (GA-BPTT) is the same as that of real-valued RNN. After the sample information is modeled by GA, it is transmitted to the output layer through the input layer to obtain the actual output, and then the loss function is used to calculate the error  $E$  between actual output and true labels. Then backpropagation corrects the weight matrix and bias parameters, respectively, until the error converges to a certain threshold.

Similar to the multi-vector activation function, the loss function of multi-vector output is essentially the loss function in the real domain for each component. In the real domain, the dynamic loss is only based on all previously connected neurons, while the difference of GA-BPTT is that the multi-vector loss is calculated for each component of the multi-vector neuron parameter, which can act as a regularizer during training.

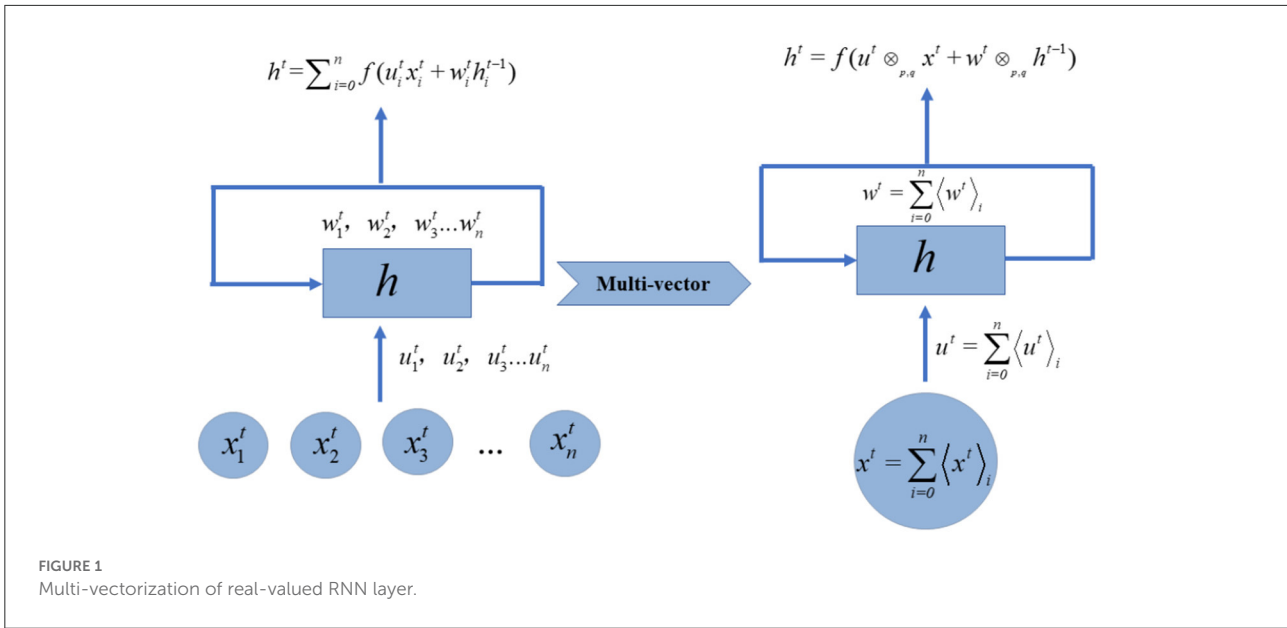
According to Equation (15), the output  $b_k^t$  can be written as

$$b_k^t = \sum_{I \in \Gamma} \left[ b_k^t \right]_I e_I \tag{17}$$

Suppose that  $y^t$  is the true labels,  $b^t$  in Equation (15) is the actual output, the final loss function is defined as the sum of the mean squared errors at each moment

$$E = \sum_{t=1}^T E^t = \frac{1}{2} \sum_{t=1}^T \|y^t - b^t\|_2 \tag{18}$$

Because the loss function is computed separately for each component of the multi-vector, the loss  $E$  is also a multi-vector.



### 3.2.1. GA-RNN output layer weight matrix

The weight matrix  $V$  of the output layer is used to calculate the actual output  $b^t$ , that is,

$$\frac{\partial E}{\partial V} = \sum_{t=1}^T \frac{\partial E^t}{\partial V}$$

$$\frac{\partial E^t}{\partial V} = \begin{pmatrix} \frac{\partial E^t}{\partial v_{11}} & \dots & \frac{\partial E^t}{\partial v_{H1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial E^t}{\partial v_{1K}} & \dots & \frac{\partial E^t}{\partial v_{HK}} \end{pmatrix} = \begin{pmatrix} \nabla E^t_{v_{11}} & \dots & \nabla E^t_{v_{H1}} \\ \vdots & \ddots & \vdots \\ \nabla E^t_{v_{1K}} & \dots & \nabla E^t_{v_{HK}} \end{pmatrix} \quad (19)$$

Each item in Equation (19) can be calculated individually, i.e.,

$$\nabla E^t_{v_{hk}} = \sum_{I \in \Gamma} \frac{\partial E^t}{\partial [v_{hk}]_I} e_I \quad (20)$$

For each component in Equation (20), the chain rule is applied for parameter updating, that is,

$$\frac{\partial E^t}{\partial [v_{hk}]_I} = \sum_{B \in \Gamma} \frac{\partial E^t}{\partial [m^t_k]_B} \cdot \frac{\partial [m^t_k]_B}{\partial [v_{hk}]_I} \quad (21)$$

The calculation of Equation (21) can be divided into the activation function part and the propagation function part, as shown in Equations (22, 23), respectively.

$$\frac{\partial E^t}{\partial [m^t_k]_B} = \frac{\partial E^t}{\partial [b^t_k]_B} \cdot \frac{\partial [b^t_k]_B}{\partial [m^t_k]_B} = ([b^t_k]_B - [y^t_k]_B) \cdot \alpha'([m^t_k]_B)$$

$$= [\delta^t_k]_B \quad (22)$$

TABLE 1 Datasets used in the experiment.

| Dataset       | Sequence length | Number of variables | Sampling interval |
|---------------|-----------------|---------------------|-------------------|
| Traffic       | 17,544          | 862                 | 1 h               |
| Electricity   | 26,304          | 321                 | 1 h               |
| Solar-Energy  | 52,560          | 173                 | 10 min            |
| Exchange-Rate | 7,588           | 8                   | 24 h              |

$$\frac{\partial [m^t_k]_B}{\partial [v_{hk}]_I} = \{R(a^t_h)\}_{(b,i)}$$

$$\frac{\partial [m^t_k]_B}{\partial [a^t_h]_B} = [v_{hk}]_0 \quad (23)$$

Where  $\{R(a^t_h)\}_{(b,i)}$  is the value in row  $b$  column  $i$  of real representation matrix  $R(a^t_h)$ . Therefore,

$$\frac{\partial E^t}{\partial [v_{hk}]_I} = \sum_{B \in \Gamma} [\delta^t_k]_B \cdot \{R(a^t_h)\}_{(b,i)} \quad (24)$$

That is,

$$\nabla E^t_{v_{jk}} = \sum_{I \in \Gamma} \left[ \sum_{B \in \Gamma} [\delta^t_k]_B \cdot \{R(a^t_h)\}_{(b,i)} \right] e_I \quad (25)$$

### 3.2.2. GA-RNN hidden layer weight matrix

The derivation process of the backpropagation for  $\mathbf{W}$  is as follows:

$$\frac{\partial E}{\partial \mathbf{W}} = \sum_{t=1}^T \frac{\partial E^t}{\partial \mathbf{W}} \quad (26)$$

Similarly,

$$\frac{\partial E^t}{\partial \mathbf{W}} = \begin{pmatrix} \frac{\partial E^t}{\partial w_{11}} & \cdots & \frac{\partial E^t}{\partial w_{H1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial E^t}{\partial w_{1H}} & \cdots & \frac{\partial E^t}{\partial w_{HH}} \end{pmatrix} = \begin{pmatrix} \nabla E^t_{w_{11}} & \cdots & \nabla E^t_{w_{H1}} \\ \vdots & \ddots & \vdots \\ \nabla E^t_{w_{1H}} & \cdots & \nabla E^t_{w_{HH}} \end{pmatrix} \quad (27)$$

Since the weights of the hidden layer are related to the state of the previous moment, consider

$$a_h^{t+1} = \sigma \left( \sum_{i=1}^N u_{ih} \otimes_{p,q} x_i^{t+1} + \sum_{h'=1}^H w_{h'h} \otimes_{p,q} a_{h'}^t + \theta_h^a \right)$$

$$b_k^{t+1} = \alpha \left( m_k^{t+1} \right) = \alpha \left( \sum_{h=1}^H v_{hk} \otimes_{p,q} a_h^{t+1} + \theta_k^b \right) \quad (28)$$

We have

$$\nabla E^t_{w_{h'h}} = \sum_{I \in \Gamma} \frac{\partial E^t}{\partial [w_{h'h}]_I} \mathbf{e}_I \quad (29)$$

In Equation (29), the calculation of each component is denoted by

$$\frac{\partial E^t}{\partial [w_{h'h}]_I} = \sum_{B \in \Gamma} \frac{\partial E^t}{\partial [m_k^t]_B} \cdot \frac{\partial [m_k^t]_B}{\partial [a_h^t]_B} \cdot \frac{\partial [a_h^t]_B}{\partial [z_h^t]_B} \cdot \frac{\partial [z_h^t]_B}{\partial [w_{h'h}]_I} \quad (30)$$

$$+ \sum_{B \in \Gamma} \frac{\partial E^t}{\partial [m_k^{t+1}]_B} \cdot \frac{\partial [m_k^{t+1}]_B}{\partial [a_h^{t+1}]_B} \cdot \frac{\partial [a_h^{t+1}]_B}{\partial [z_h^{t+1}]_B} \cdot \frac{\partial [z_h^{t+1}]_B}{\partial [w_{h'h}]_I}$$

Where

$$\sum_{B \in \Gamma} \frac{\partial E^t}{\partial [m_k^t]_B} \cdot \frac{\partial [m_k^t]_B}{\partial [a_h^t]_B} \cdot \frac{\partial [a_h^t]_B}{\partial [z_h^t]_B} \cdot \frac{\partial [z_h^t]_B}{\partial [w_{h'h}]_I}$$

$$= \sum_{B \in \Gamma} [\delta_k^t]_B \cdot [v_{hk}]_0 \cdot \sigma' \left( [z_h^t]_B \right) \cdot \left\{ \mathbf{R} \left( a_h^{t-1} \right) \right\}_{(b,i)}$$

$$+ \sum_{B \in \Gamma} \frac{\partial E^t}{\partial [m_k^{t+1}]_B} \cdot \frac{\partial [m_k^{t+1}]_B}{\partial [a_h^{t+1}]_B} \cdot \frac{\partial [a_h^{t+1}]_B}{\partial [z_h^{t+1}]_B} \cdot \frac{\partial [z_h^{t+1}]_B}{\partial [w_{h'h}]_I}$$

$$= \sum_{B \in \Gamma} [\delta_k^{t+1}]_B \cdot [v_{hk}]_0 \cdot \sigma' \left( [z_h^{t+1}]_B \right) \cdot \left\{ \mathbf{R} \left( a_h^t \right) \right\}_{(b,i)} \quad (31)$$

### 3.2.3. GA-RNN input layer weight matrix

The updating of input layer weight matrix  $\mathbf{U}$  is the same as that of the hidden layer, that is,

$$\frac{\partial E}{\partial \mathbf{U}} = \sum_{t=1}^T \frac{\partial E^t}{\partial \mathbf{U}}$$

$$\frac{\partial E^t}{\partial \mathbf{U}} = \begin{pmatrix} \frac{\partial E^t}{\partial u_{11}} & \cdots & \frac{\partial E^t}{\partial u_{N1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial E^t}{\partial u_{1H}} & \cdots & \frac{\partial E^t}{\partial u_{NH}} \end{pmatrix} = \begin{pmatrix} \nabla E^t_{u_{11}} & \cdots & \nabla E^t_{u_{N1}} \\ \vdots & \ddots & \vdots \\ \nabla E^t_{u_{1H}} & \cdots & \nabla E^t_{u_{NH}} \end{pmatrix} \quad (32)$$

Where

$$\nabla E^t_{u_{ih}} = \sum_{I \in \Gamma} \frac{\partial E^t}{\partial [u_{ih}]_I} \mathbf{e}_I$$

$$\frac{\partial E^t}{\partial (u_{ih})_I} = \sum_{B \in \Gamma} \frac{\partial E^t}{\partial [m_k^t]_B} \cdot \frac{\partial [m_k^t]_B}{\partial [a_h^t]_B} \cdot \frac{\partial [a_h^t]_B}{\partial [z_h^t]_B} \cdot \frac{\partial [z_h^t]_B}{\partial [u_{ih}]_I}$$

$$= \sum_{B \in \Gamma} [\delta_k^t]_B \cdot [v_{hk}]_0 \cdot \sigma' \left( [z_h^t]_B \right) \cdot \left\{ \mathbf{R} \left( x_i^t \right) \right\}_{(b,i)} \quad (33)$$

That is,

$$\nabla E^t_{u_{ih}} = \sum_{I \in \Gamma} \left[ \sum_{B \in \Gamma} [\delta_k^t]_B \cdot [v_{hk}]_0 \cdot \sigma' \left( [z_h^t]_B \right) \cdot \left\{ \mathbf{R} \left( x_i^t \right) \right\}_{(b,i)} \right] \mathbf{e}_I \quad (34)$$

### 3.2.4. GA-RNN output layer bias

$\frac{\partial E}{\partial \theta^b}$  can be written as

$$\frac{\partial E}{\partial \theta^b} = \sum_{t=1}^T \frac{\partial E^t}{\partial \theta^b} = \sum_{t=1}^T \begin{pmatrix} \frac{\partial E^t}{\partial \theta_1^b} \\ \vdots \\ \frac{\partial E^t}{\partial \theta_K^b} \end{pmatrix} = \sum_{t=1}^T \begin{pmatrix} \nabla E^t_{\theta_1^b} \\ \vdots \\ \nabla E^t_{\theta_K^b} \end{pmatrix} \quad (35)$$

Similarly,

$$\nabla E^t_{\theta_k^b} = \sum_{I \in \Gamma} \frac{\partial E^t}{\partial [\theta_k^b]_I} \mathbf{e}_I$$

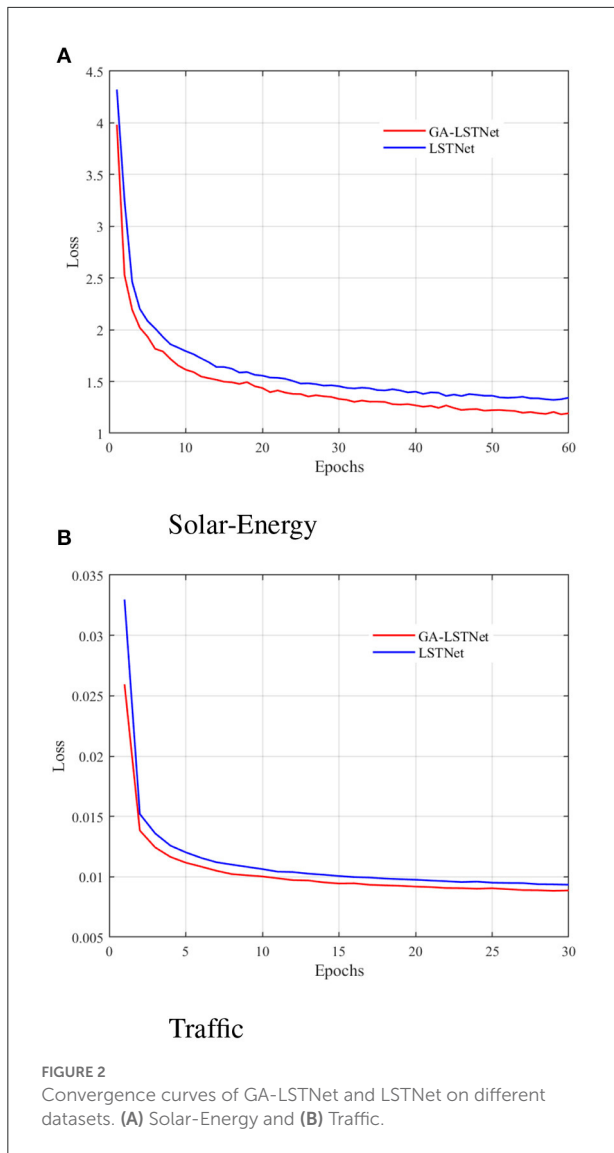
$$\frac{\partial E^t}{\partial [\theta_k^b]_I} = \sum_{B \in \Gamma} \frac{\partial E^t}{\partial [m_k^t]_B} \cdot \frac{\partial [m_k^t]_B}{\partial [\theta_k^b]_I} \quad (36)$$

$$= \sum_{B \in \Gamma} [\delta_k^t]_B \cdot \frac{\partial [m_k^t]_B}{\partial [\theta_k^b]_I} = [\delta_k^t]_I$$

Therefore,

$$\nabla E^t_{\theta_k^b} = \sum_{I \in \Gamma} [\delta_k^t]_I \mathbf{e}_I \quad (37)$$





### 3.2.5. GA-RNN hidden layer bias

Same as Equations (35, 36):

$$\nabla E_{\theta_h^a}^t = \sum_{I \in \Gamma} \frac{\partial E^t}{\partial [\theta_h^a]_I} e_I \quad (38)$$

The difference is that the weight updating of the hidden layer takes into account the state of the previous moment, namely:

$$\begin{aligned} \frac{\partial E^t}{\partial [\theta_h^a]_I} &= \sum_{B \in \Gamma} \frac{\partial E^t}{\partial [m_k^t]_B} \cdot \frac{\partial [m_k^t]_B}{\partial [a_h^t]_B} \cdot \frac{\partial [a_h^t]_B}{\partial [z_h^t]_B} \cdot \frac{\partial [z_h^t]_B}{\partial [\theta_h^a]_I} \\ &= \sum_{B \in \Gamma} [\delta_k^t]_B \cdot [v_{hk}]_0 \cdot \sigma' \left( [z_h^t]_B \right) \cdot \frac{\partial [z_h^t]_B}{\partial [\theta_h^a]_I} \\ &= [\delta_k^t]_I \cdot [v_{hk}]_0 \cdot \sigma' \left( [z_h^t]_I \right) \end{aligned} \quad (39)$$

Therefore,

$$\nabla E_{\theta_h^a}^t = \sum_{I \in \Gamma} [\delta_k^t]_I \cdot [v_{hk}]_0 \cdot \sigma' \left( [z_h^t]_I \right) e_I \quad (40)$$

## 3.3. Geometric algebra based long short-term memory network layer

Due to the “long-term dependence” problem caused by the disappearance of the RNN gradient, LSTM emerged. It has the ability to learn long-distance dependencies. This mechanism can be easily extended to the GA (GA-LSTM). Gates of LSTM is the core components, and the GA gates are characterized by the fusion process of each component of the multi-vector input signal after multiplication with the components of the multi-vector gate parameters. Let  $f^t$ ,  $i^t$ ,  $o^t$ ,  $c^t$ , and  $h^t$  be the forget gate, input gate, output gate, cell state and hidden state of the GA-LSTM unit at time step  $t$ , respectively. Then the GA-LSTM propagation process can be defined as:

$$\begin{aligned} f^t &= \sigma \left( U_f \otimes_{p,q} x^t + W_f \otimes_{p,q} h^{t-1} + \theta^f \right) \\ i^t &= \sigma \left( U_i \otimes_{p,q} x^t + W_i \otimes_{p,q} h^{t-1} + \theta^i \right) \\ c^t &= f^t \odot c^{t-1} + i^t \odot \alpha \left( U_c \otimes_{p,q} x^t + W_c \otimes_{p,q} h^{t-1} + \theta^c \right) \\ o^t &= \sigma \left( U_o \otimes_{p,q} x^t + W_o \otimes_{p,q} h^{t-1} + \theta^o \right) \\ h^t &= o^t \odot \alpha \left( c^t \right) \end{aligned} \quad (41)$$

## 4. Results and discussion

### 4.1. Datasets

This experiment uses four publicly available datasets, namely Traffic, Electricity, Solar-Energy and Exchange-Rate. As shown in Table 1, Traffic records the occupancy rates (0–1) measured by simultaneous interpreting 862 different sensors on the San Francisco Bay expressway within 2 years (2015–2016 years), and the data are collected once 1 h. Electricity records the power consumption of 321 customers from 2012 to 2014. The data is collected every 15 min. This part converts the data to reflect the hourly consumption; Solar-Energy is the solar power generation record of 137 photovoltaic power stations in Alabama in 2006, which is sampled every 10 min. Exchange-Rate is the summary of daily exchange rates of eight countries including Australia, the United Kingdom, Canada, Switzerland, China, Japan, New Zealand and Singapore from 1990 to 2016. These datasets are real-world data and contain linear and nonlinear interdependencies (Jordan et al., 2003). All datasets are divided into training set (60%), verification set (20%) and test set (20%) in chronological order. The download address of the four datasets is here.

TABLE 2 Prediction results using RAE as indicator.

| Dataset       | Horizon | AR            | LSVR          | LRidge        | TRMF   | GP     | LSTNet        | GA-LSTNet     |
|---------------|---------|---------------|---------------|---------------|--------|--------|---------------|---------------|
| Traffic       | 3       | 0.4491        | 0.4929        | 0.4965        | 0.5823 | 0.5148 | 0.3238        | <b>0.311</b>  |
|               | 6       | 0.461         | 0.5483        | 0.5115        | 0.4653 | 0.5759 | 0.3254        | <b>0.3144</b> |
|               | 12      | 0.47          | 0.7454        | 0.5198        | 0.4868 | 0.5316 | 0.3433        | <b>0.3308</b> |
|               | 24      | 0.4696        | 0.4761        | 0.4846        | 0.5352 | 0.4829 | 0.354         | <b>0.3357</b> |
| Electricity   | 3       | 0.0579        | 0.0858        | 0.09          | 0.1032 | 0.0907 | 0.0493        | <b>0.0485</b> |
|               | 6       | 0.0598        | 0.0816        | 0.0933        | 0.1269 | 0.1137 | <b>0.0526</b> | 0.0541        |
|               | 12      | 0.0603        | 0.0762        | 0.1268        | 0.1328 | 0.1043 | <b>0.0534</b> | 0.0561        |
|               | 24      | 0.0611        | 0.069         | 0.0779        | 0.201  | 0.0776 | <b>0.0545</b> | 0.0593        |
| Solar-Energy  | 3       | 0.1846        | <b>0.1082</b> | 0.1227        | 0.1326 | 0.1419 | 0.1462        | 0.1083        |
|               | 6       | 0.3242        | 0.2451        | 0.2098        | 0.1984 | 0.2189 | 0.1462        | <b>0.1315</b> |
|               | 12      | 0.5673        | 0.4362        | 0.407         | 0.4786 | 0.4095 | <b>0.1917</b> | 0.1959        |
|               | 24      | 0.9221        | 0.618         | 0.6977        | 0.9527 | 0.7599 | 0.3042        | <b>0.2542</b> |
| Exchange-Rate | 3       | 0.0181        | 0.0148        | <b>0.0144</b> | 0.0289 | 0.023  | 0.0364        | 0.0189        |
|               | 6       | 0.0224        | 0.0231        | 0.0225        | 0.0517 | 0.0239 | <b>0.0207</b> | 0.025         |
|               | 12      | 0.0291        | 0.036         | 0.0358        | 0.0429 | 0.0355 | <b>0.0286</b> | 0.0323        |
|               | 24      | <b>0.0378</b> | 0.0571        | 0.0602        | 0.058  | 0.0547 | 0.0475        | 0.0505        |

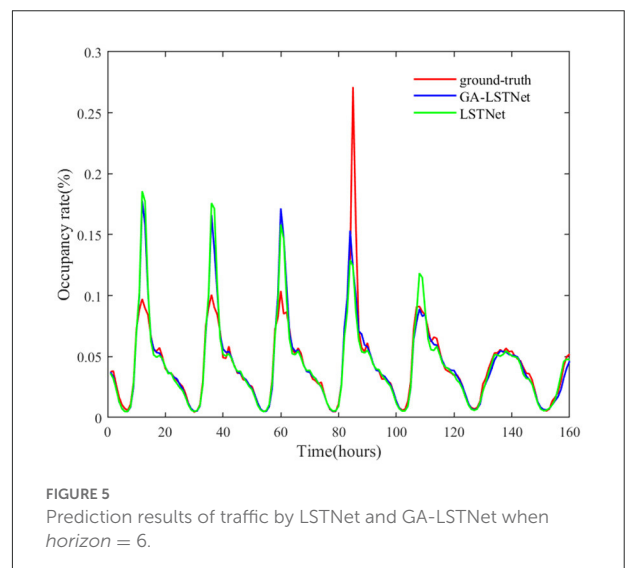
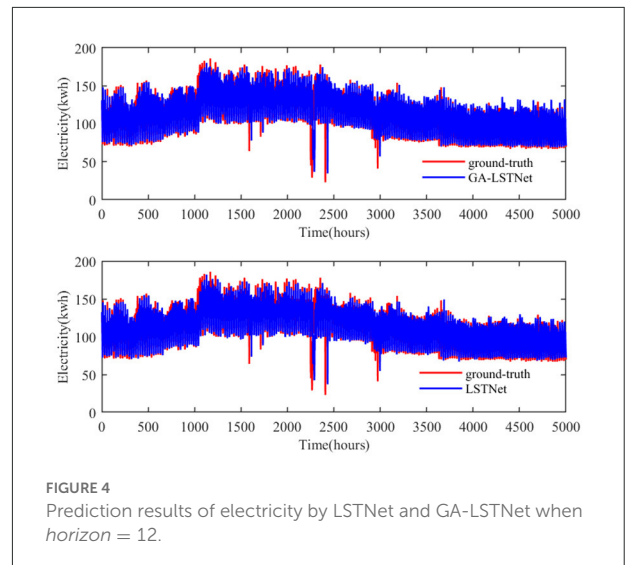
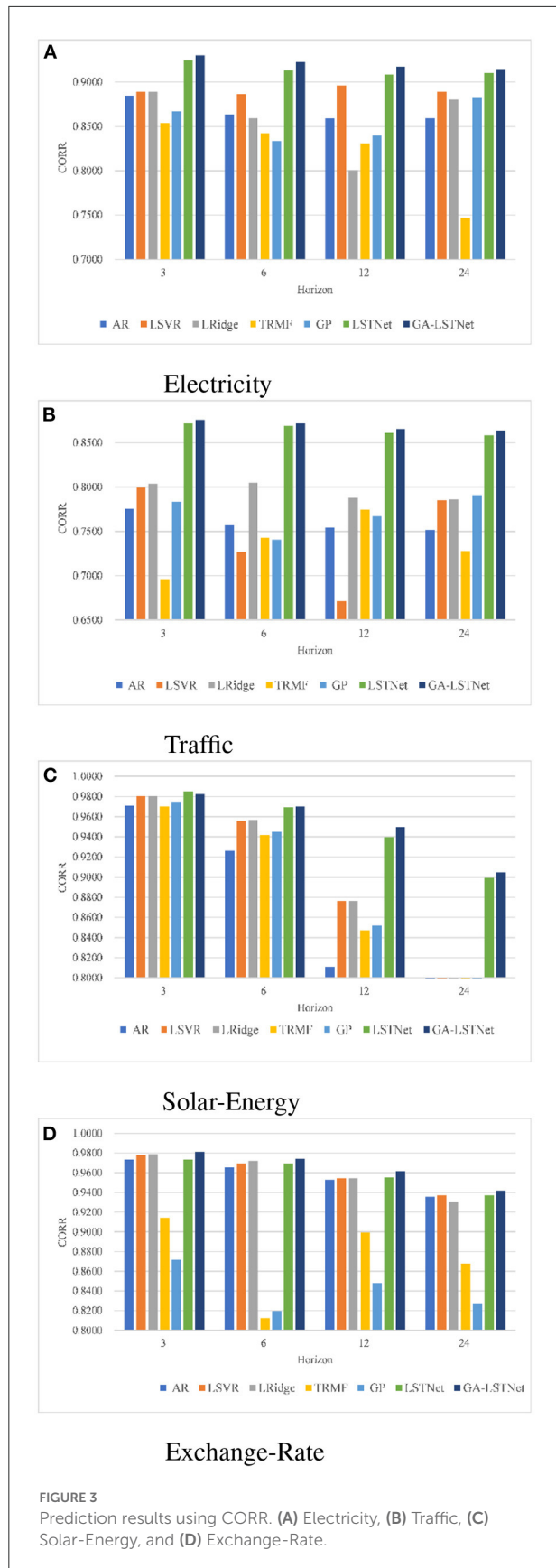
The bold values indicate the best RAE and RSE of the prediction results of the seven methods.

TABLE 3 Prediction results using RSE as indicator.

| Dataset       | Horizon | AR            | LSVR   | LRidge        | TRMF   | GP     | LSTNet        | GA-LSTNet     |
|---------------|---------|---------------|--------|---------------|--------|--------|---------------|---------------|
| Traffic       | 3       | 0.5991        | 0.574  | 0.5833        | 0.6708 | 0.6082 | 0.4896        | <b>0.4798</b> |
|               | 6       | 0.6218        | 0.658  | 0.592         | 0.6261 | 0.6772 | 0.4909        | <b>0.4852</b> |
|               | 12      | 0.6252        | 0.7714 | 0.6148        | 0.5956 | 0.6406 | 0.507         | <b>0.5002</b> |
|               | 24      | 0.6293        | 0.5909 | 0.6025        | 0.6442 | 0.5995 | 0.5188        | <b>0.5026</b> |
| Electricity   | 3       | 0.0995        | 0.1523 | 0.1467        | 0.1802 | 0.15   | 0.0851        | <b>0.0844</b> |
|               | 6       | 0.1035        | 0.1372 | 0.1419        | 0.2039 | 0.1907 | 0.0941        | <b>0.0935</b> |
|               | 12      | 0.105         | 0.1333 | 0.2129        | 0.2189 | 0.1621 | <b>0.0967</b> | 0.0975        |
|               | 24      | 0.1054        | 0.118  | 0.128         | 0.3656 | 0.1273 | <b>0.1046</b> | 0.1101        |
| Solar-Energy  | 3       | 0.2435        | 0.2021 | 0.2019        | 0.2473 | 0.2259 | 0.2067        | <b>0.1976</b> |
|               | 6       | 0.379         | 0.2999 | 0.2954        | 0.347  | 0.3286 | 0.2606        | <b>0.2462</b> |
|               | 12      | 0.5911        | 0.4846 | 0.4832        | 0.5597 | 0.52   | 0.3563        | <b>0.3445</b> |
|               | 24      | 0.8699        | 0.73   | 0.7287        | 0.9005 | 0.7973 | 0.4695        | <b>0.4306</b> |
| Exchange-Rate | 3       | 0.0228        | 0.0189 | <b>0.0184</b> | 0.0351 | 0.0239 | 0.0431        | 0.0234        |
|               | 6       | 0.0279        | 0.0284 | 0.0274        | 0.0875 | 0.0272 | <b>0.0256</b> | 0.0297        |
|               | 12      | 0.0353        | 0.0425 | 0.0419        | 0.0494 | 0.0394 | <b>0.0352</b> | 0.0401        |
|               | 24      | <b>0.0445</b> | 0.0662 | 0.0675        | 0.0653 | 0.058  | 0.0475        | 0.0594        |

The bold values indicate the best RAE and RSE of the prediction results of the seven methods.





## 4.2. Experimental design

In order to verify the performance of the proposed neural networks based on GA, several MTS prediction algorithms with the proposed network are implemented, which are:

- 1) AR: Standard autoregressive model,
- 2) LSVR (Li and Cyrus, 2018): Vector autoregressive model with support vector regression objective function,
- 3) LRidge: Vector autoregressive model with L2 regularization,
- 4) TRMF (Hsiang et al., 2016): Autoregressive model of time regularization matrix decomposition,
- 5) GP (Roberts et al., 2012): Gaussian process for time-series modeling,

6) LSTNet (Lai et al., 2018): Deep neural network for modeling long-term and short-term time patterns,

7) GA-LSTNet: A geometric algebra based depth neural network for modeling long-term and short-term time patterns proposed in this paper.

For the first six comparison methods, the parameter setting used in this experiment is the same as that of Lai et al. (2018). That is, grid search is performed for all adjustable hyperparameters on the verification set of each method and each dataset. Specifically, the regularization coefficient of AR is selected from {0.1, 1, 10} to achieve the best performance. The value range of LSVR and LRidge regularization coefficients is  $\{2^{-10}, 2^{-9}, \dots, 2^9, 2^{10}\}$ . TRME, the search ranges of hidden dimension and regularization coefficient are  $\{2^2, 2^3, \dots, 2^6\}$  and {0.1, 1, 10}, respectively.

The GA-LSTNet used in this paper is composed of convolution layer and LSTM of GA instead of convolution layer and LSTM of LSTNet. LSTNet and GA-LSTNet have the same network structure and parameter settings. Their differences are the form of data input and the calculation method of network layer. Specifically, the selection range of hidden dimensions of LSTM and convolution layer is {50, 100, 200} and {20, 50, 100}. For the number of RNN-skip layers, Electricity and Traffic are set to 24, and the adjustment range of Solar-Energy and Exchange-Rate is  $2^1-2^6$ . In addition to the input and output layers, dropout with a size of 0.1 or 0.2 is set after each layer. The optimizer for the two models is Adam.

In order to quantify and represent all experimental results, the seven methods in this section follow the same evaluation index (Hsiang et al., 2016): relative absolute error (RAE), root relative square error (RSE) and correlation coefficient (CORR). The first evaluation criterion RAE is defined as:

$$RAE = \frac{\sqrt{\sum_{t=t_0}^{t_1} \sum_{i=1}^n |(y_{t,i} - \hat{y}_{t,i})|}}{\sqrt{\sum_{t=t_0}^{t_1} \sum_{i=1}^n |\hat{y}_{t,i} - \hat{y}_{t_0:t_1,1:n}|}} \quad (42)$$

The second evaluation criterion RSE is defined as:

$$RSE = \frac{\sqrt{\sum_{t=t_0}^{t_1} \sum_{i=1}^n (y_{t,i} - \hat{y}_{t,i})^2}}{\sqrt{\sum_{t=t_0}^{t_1} \sum_{i=1}^n (\hat{y}_{t,i} - \hat{y}_{t_0:t_1,1:n})^2}} \quad (43)$$

The third evaluation criterion CORR is defined as:

$$CORR = \frac{1}{n} \sum_{i=1}^n \frac{\sum_{t=t_0}^{t_1} (y_{t,i} - \bar{y}_{t_0:t_1,i}) (\hat{y}_{t,i} - \bar{\hat{y}}_{t_0:t_1,i})}{\sqrt{\sum_{t=t_0}^{t_1} (y_{t,i} - \bar{y}_{t_0:t_1,i})^2} \sqrt{\sum_{t=t_0}^{t_1} (\hat{y}_{t,i} - \bar{\hat{y}}_{t_0:t_1,i})^2}} \quad (44)$$

Where  $y$  is the predicted value,  $\hat{y}$  is the real value of the test set,  $\bar{y}$  represents the mean value of the set, and  $t \in [t_0, t_1]$  is the data label of the test set. For RAE and RSE, the lower the value, the better the prediction result. On the contrary, for CORR, the higher the value, the better the prediction result.

### 4.3. Experimental results and analysis

In this part, seven methods will be used to conduct prediction experiments on four datasets, and the prediction range horizon is set to {3, 6, 12, 24}. According to Table 1, for Electricity and Traffic, the prediction range is set to {3, 6, 12, 24} h. For Solar-Energy, the prediction range is set to {30, 60, 120, 240} min. The prediction range for Exchange-Rate is set to {3, 6, 12, 24} days.

This part first compares the convergence curves of the two networks on Traffic and Solar-Energy. Figure 2A shows the convergence curve of GA-LSTNet and LSTNet predicting power generation in the next 240 min on the Solar-Energy dataset; Figure 2B shows the convergence curve of GA-LSTNet and LSTNet predicting traffic flow in the next 6 h on the Traffic dataset. The curves of the two experiments show that under the same network configuration, the convergence rate of GA-LSTNet is faster than LSTNet, and the final convergence value and the training error are smaller.

Tables 2, 3, respectively, show the comparison of RAE and RSE of the prediction results of the seven methods, in which the best results are shown in bold.

It can be seen from the results in Tables 2, 3 that on the Traffic and Solar-Energy datasets, the GA-LSTNet model proposed in this part has absolute advantages over the LSTNet model with the same structure and achieves lower prediction error. On the Electricity dataset, although LSTNet achieved the best results in some predictions, it can be observed that the difference between GA-LSTNet and LSTNet is no more than 0.002, which is a very small gap. The experimental results show the feasibility of GA-LSTNet, because it can capture more useful information in different data channels when it represents MTS as GA multi-vector. Therefore, the LSTM based on GA has better performance than the real-valued LSTM in MTS prediction.

In addition, in order to more intuitively observe the prediction results, Figure 3 shows the prediction results of seven different methods under different conditions using CORR as an index. In Figure 3, the higher the bar chart, the greater the CORR representing the corresponding prediction task and the higher the prediction accuracy. As shown in Figure 3, GA-LSTNet obtained the highest CORR value in each prediction task of Electricity, Traffic and Exchange-Rate. In the prediction task of horizon = 3 for Solar-Energy dataset, the accuracy is slightly lower than LSTNet, but with the growth of prediction time, the superiority of GA-LSTNet is gradually obvious. It can be seen that the introduction of GA into real-valued RNN and LSTM will not change their basic properties. The capture of the correlation between different features by GA enhances the long-term dependent learning ability of real-valued neural networks.

From the above quantitative results, it can be seen that the overall accuracy of GA-LSTNet is improved compared with LSTNet, which means that the predicted positions of some points will be more consistent with the real labels. As shown in Figures 4, 5, two networks are selected to visually observe all the prediction results of Electricity and some prediction results on Traffic dataset.

In Figure 4, red represents the real value of power consumption and blue represents the predicted value. GA-LSTNet and LSTNet are similar to the real value trend on the whole, but GA-LSTNet performs better in details, especially in some prominent troughs. For example, in 2,000–2,500 h, the gap between GA-LSTNet and the real value is smaller and closer to the real value.

In Figure 5, the red curve represents the real value of Traffic occupancy within 160 h of the test set, the green curve represents the predicted value of LSTNet, and the blue curve represents the predicted value of GA-LSTNet. Both are basically consistent with the real value at the trough, but among the six peaks shown in Figure 5, five are GA-LSTNet, which is closer to the real value. It can be seen from the comparison figure that after replacing the corresponding real-valued network with GA convolution and LSTM, GA-LSTNet retains more useful relevant information in the original data due to the multi-dimensional consistency of GA, and the prediction performance is significantly improved.

## 5. Conclusion

This paper focuses on the construction of the geometric algebra based RNN and LSTM for the processing of MTS. Under the framework of GA, the MTS is encoded into GA multi-vectors to avoid the loss of structural relationships among multi-dimensional variables. And then the forward and backpropagation algorithms for the proposed GA-RNN, GA-LSTM are derived. The experimental results show that the GA-LSTNet has good convergence and more accurate prediction accuracy in MTS prediction, and has certain advantages compared with the real-valued LSTNet. GA-RNN and GA-LSTM provide a more accurate solution for the existing shortcomings of MTS prediction models. At next steps, we will focus on more practical MTS applications with the proposed networks.

## References

Ai, G., Chang, W. C., and Yang, Y. (2017). Modeling long- and short-term temporal patterns with deep neural networks. *arXiv:1703.07015 [cs.LG]*. doi: 10.48550/arXiv.1703.07015

## Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding authors.

## Author contributions

YL and YueW proposed the new idea of the paper and participated in the outage performance analysis. YiW performed the simulations and drafted the paper. CQ played an important role in interpreting the results and revised the manuscript. RW conceived of the study, participated in its design, coordination, and helped to draft the manuscript. All authors have read and approved the final manuscript.

## Funding

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant Nos. 61771299 and 62071286, the Clinical Research Funds for Shanghai Municipal Health Commission (202040170).

## Acknowledgments

We thank reviewers for constructive feedback on the manuscript, all participants who participated in the study.

## Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

## Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

Bai, S., Kolter, J. Z., and Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv:1803.01271 [cs.LG]*. doi: 10.48550/arXiv.1803.01271

- Cao, L. J., and Tay, F. E. H. (2003). Support vector machine with adaptive parameters in financial time series forecasting. *IEEE Trans. Neural Netw.* 14, 1506–1518. doi: 10.1109/TNN.2003.820556
- Chen, R., Xiao, H., and Yang, D. (2021). Autoregressive models for matrix-valued time series. *J. Econometr.* 222(1, Part B), 539–560. doi: 10.1016/j.jeconom.2020.07.015
- Cho, K., Merriënboer, B. V., and Bahdanau, D. (2014). “On the properties of neural machine translation: encoder-decoder approaches,” in *Proceedings of SSSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation* (Doha: Association for Computational Linguistics), 103–111.
- Dauphin, Y. N., Fan, A., and Auli, M. (2017). “Language modeling with gated convolutional networks,” in *Proceedings of the 34th International Conference on Machine Learning, Volume 70 of ICML'17* (Sydney, NSW), 933–941. Available online at: <https://www.jmlr.org>
- Faloutsos, C., Gasthaus, J., Januschowski, T., and Wang, Y. (2018). Forecasting big time series: old and new. *Proc. VLDB Endowment* 11, 2102–2105. doi: 10.14778/3229863.3229878
- Grassi, F., Loukas, A., and Perraudin, N. (2018). A time-vertex signal processing framework: scalable processing and meaningful representations for time-series on graphs. *IEEE Trans. Signal Process.* 66, 817–829. doi: 10.1109/TSP.2017.2775589
- Gurland, J., and Whittle, P. (1951). Hypothesis testing in time series analysis. *J. Am. Stat. Assoc.* 49, 197–199. doi: 10.2307/2281054
- Han, F., Lu, H., and Liu, H. (2015). A direct estimation of high dimensional stationary vector autoregressions. *J. Mach. Learn. Res.* 16, 3115–3150. doi: 10.48550/arXiv.1307.0293
- Hestenes, D. (1986). *New Foundations for Classical Mechanics*. New York, NY: Kluwer Academic Publishers.
- Hochreiter, S., and Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.* 9, 1735–1780. doi: 10.1162/neco.1997.9.8.1735
- Hsiang, F. Y., Nikhil, R., and Dhillon, I. S. (2016). “Temporal regularized matrix factorization for high-dimensional time series prediction,” in *NIPS'16: Proceedings of the 30th International Conference on Neural Information Processing Systems, NIPS'16* (Red Hook, NY: Curran Associates Inc.), 847–855.
- Isufi, E., Loukas, A., and Perraudin, N. (2019). Forecasting time series with varma recursions on graphs. *IEEE Trans. Signal Process.* 67, 4870–4885. doi: 10.1109/TSP.2019.2929930
- Jordan, N., Becker, R., and Gunsolus, J. (2003). Knowledge networks: an avenue to ecological management of invasive weeds. *Weed Sci.* 51, 271–277. doi: 10.1614/0043-1745(2003)051[0271:KNAATE]2.0.CO;2
- Lai, G., Chang, W. C., and Yang, Y. (2018). “Modeling long- and short-term temporal patterns with deep neural networks,” in *SIGIR '18: The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '18* (New York, NY: Association for Computing Machinery), 95–104.
- Li, Y., Yu, R., and Shahabi, C. (2017). Diffusion convolutional recurrent neural network: data-driven traffic forecasting. *arXiv:1707.01926 [cs.LG]*. doi: 10.48550/arXiv.1707.01926
- Li, Y. G., and Cyrus, S. (2018). A brief overview of machine learning methods for short-term traffic forecasting and future directions. *SIGSPATIAL Special* 10, 3–9. doi: 10.1145/3231541.3231544
- López-González, G., Altamirano-Gómez, G., and Bayro-Corrochano, E. (2016). Geometric entities voting schemes in the conformal geometric algebra framework. *Adv. Appl. Clifford Algebras* 26, 1045–1059. doi: 10.1007/s00006-015-0589-y
- Loukas, A., and Perraudin, N. (2019). Stationary time-vertex signal processing. *EURASIP J. Adv. Signal Process* 36, 1–19. doi: 10.1186/s13634-019-0631-7
- Mohammad, Z., Hossein, B., and Isa, E. (2019). A reliable linear stochastic daily soil temperature forecast model. *Soil Tillage Res.* 189, 73–87. doi: 10.1016/j.still.2018.12.023
- Parcollet, T., Ravanelli, M., and Morchid, M. (2019). “Bidirectional quaternion long-short term memory recurrent neural networks for speech recognition,” in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Brighton: IEEE), 8519–8523.
- Rafal, A. (2004). *Lectures on Clifford (Geometric) Algebras and Applications*. Boston, MA: Birkhäuser.
- Roberts, S., Osborne, M., and Ebdon, M. (2012). Gaussian processes for time-series modelling. *Philos. Trans.* 371, 20110550. doi: 10.1098/rsta.2011.0550
- Roy, S. K., Krishna, G., and Dubey, S. R. (2020). Hybridsn: exploring 3-d-2-d cnn feature hierarchy for hyperspectral image classification. *IEEE Geosci. Remote Sens. Lett.* 17, 277–281. doi: 10.1109/LGRS.2019.2918719
- Sen, R., Yu, H. F., and Dhillon, I. (2019). Think globally, act locally: a deep neural network approach to high-dimensional time series forecasting. *arXiv:1905.03806 [stat.ML]*. doi: 10.48550/arXiv.1905.03806
- Syama, S. R., Matthias, W. S., and Jan, G. (2018). “Deep state space models for time series forecasting,” in *NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS'18* (Red Hook, NY: Curran Associates Inc.), 7796–7805.
- Wu, Z. H., Pan, S. R., and Guo, D. L. (2019). “Graph wavenet for deep spatial-temporal graph modeling,” in *IJCAI'19: Proceedings of the 28th International Joint Conference on Artificial Intelligence, IJCAI'19* (Macao: AAAI Press), 1907–1913.
- Yu, B., Yin, H., and Zhu, Z. (2018). “Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting,” in *IJCAI'18: Proceedings of the 27th International Joint Conference on Artificial Intelligence, IJCAI'18* (Stockholm: AAAI Press), 3634–3640.
- Zhang, L., Aggarwal, C., and Qi, G. J. (2017). “Stock price prediction via discovering multi-frequency trading patterns,” in *KDD '17: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '17* (New York, NY: Association for Computing Machinery), 2141–2149.