Cuneate recurrent neural network

Lingkai Hu[a], Feng Zhan[b], Wenkai Huang[a,*], Weiming Gan[a], Haoxiang Hu[c]

[a]School of Mechanical and Electrical Engineering, Guangzhou University, Guangzhou, 510006, China
[b]The Hong Kong University of Science and Technology (Guangzhou), Smart Manufacturing Thrust, Nansha, Guangzhou, 511400, China
[c]College of Information Science and Technology, Beijing University of Chemical Technology, Beijing, 102202, China

ARTICLE INFO

Keywords:
Fault diagnosis
Long-term memory
Filtering interference
Recurrent Neural Network
Information fusion applications

ABSTRACT

Recurrent neural network (RNN) is a neural network with temporal memory capability, and this structure is widely applied in the analysis and operation of sequential data. However, due to the limitations of the chain structure of RNN, the effective information is easily disturbed by the updated information in the state update, which leads to a low signal-to-noise ratio and memory decline of the hidden state chain and affects the stability of the gradient in neural network optimization. At present, most optimization schemes for RNN can only mitigate the negative effects of the above-mentioned defects by adding gated update mechanisms or residual connections, which still lead to training failure when facing long-sequence problems because of the inability to establish long-term dependencies. Inspired by the mechanism of mammalian brain cuneate circuit organization for filtering interference signals, this paper proposes a novel cuneate recurrent neural network (CRNN), which can filter invalid information layer by layer and integrate effective information through the cuneate layer structure to realize the modeling of long-term dependencies in sequences. Through several long-sequence tasks, we demonstrate that the CRNN proposed in this paper can achieve long-term memory of valid information, and significantly improve optimization accuracy and speed.

1. Introduction

Sequence analysis is crucial research in artificial neural networks [1, 2], covering problems such as text translation [3–5], weather prediction [6, 7], speech recognition [8], brainwave analysis [9, 10], and fault signal detection [11, 12]. Various neural network structures have been designed to solve the sequence analysis tasks, including recurrent neural networks (RNN) and its variants [13–15], which have achieved great success in short sequence tasks [16]. However, many missions for analyzing long sequence data, such as music generation [17], music modeling [18], long-form translation, concrete stress analysis [19, 20], and remaining useful life prediction [21–23], are challenging for current sequence analysis frameworks. This is due to their difficulty in balancing the establishment of long-term dependencies, reducing the number of parameters, and maintaining sequence continuity [24, 25]. The essence of this problem is that artificial neural networks are susceptible to interference from updated information in state updates, leading to low signal-to-noise ratios (SNR) and memory decline in hidden state chains, which affects the stability of gradients in neural network training [26]. In addition, since the information propagation path grows with sequence length, this defect is amplified in long-sequence tasks, leading to difficulties in establishing long-term dependencies. Therefore, solving the above problems is currently a great challenge in the field of sequence analysis.

In general, human cognition of the unknown is a multi-level, low-to-high-level process. Summarizing the more abstract features from the simple ones and making judgments is a necessary process for cognition [27]. However, RNN and its variants contradict this cognitive approach by processing low-level features in sequential order and storing high-level features in the hidden states of the memory, updating the state information with each low-level feature processed. This causes the memory in the neural network to be constantly disturbed by subsequent additions of updated information, making it difficult to remember high-level features over time and leading to difficulties in making judgments based on long-term dependencies [16, 28].

In this paper, an enlightening solution is obtained from a neuroscientific perspective [29]: the tactile feedback signals from the mammalian limb do not all ascend to the sensorimotor cortex but are heavily targeted at inhibitory cuneate neurons (Fig. 1A). The cuneate neurons selectively enhance the received information, inhibit the rest, and project the filtered information to several subcortical targets. This center-surround inhibition could sharpen receptive

*Corresponding author.
E-mail address: smallkat@gzhu.edu.cn (Wenkai Huang)

fields to augment resolution and acuity. In principle, the amplification of tactile feedback could improve discrimination and select the inputs signal most relevant to the animal's behavior, and invading feedback signals could be filtered out by this feedback mechanism to enable successful animal behavior.

The same mammalian neural signal filtering mechanism can be applied to artificial neural networks. Based on the operation of the mammalian cuneate circuit organization in the above study, this paper proposes a cuneate recurrent neural network (CRNN), a structure that can extremely improve the optimization efficiency of RNNs in long-sequence tasks. The is a combination of multiple layers of RNN, and it uses special cuneate layers between the upper and lower RNN layers (Fig. 1B). The cuneate layers are designed concerning the signal filtering mechanism of mammalian cuneate neurons, where the output features of each recurrent layer are selectively signal-filtered and fed into the next higher recurrent layer. The cuneate layer can perform signal filtering for lower-level information and shorten the information propagation chain to prevent the loss and interference of effective information in recursive operations, ensuring that the neural network can perform effective long-term dependency analysis.

In summary, this paper proposes a CRNN inspired by the operation mechanism of mammalian cuneate circuit organization. The proposed network greatly improves the optimization speed of RNNs, solves the problem of memory decline of traditional RNNs in ultralong-sequence tasks [24, 30], and has high generalizability in various sequence analysis tasks.

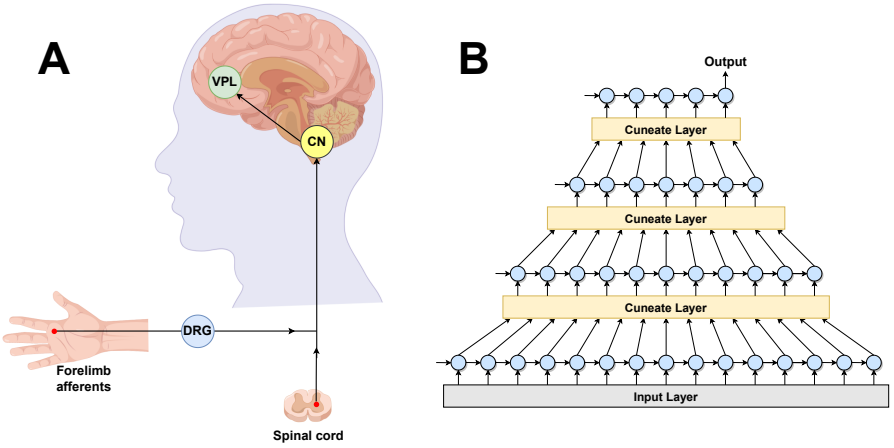

Fig. 1. (A) Cuneate nucleus (CN) receives input from forelimb afferents in the dorsal root ganglia (DRG) and from ascending projections from the spinal cord. These signals are filtered by cuneate neurons and then project to several specific subcortical targets, most predominately the ventral posterolateral nucleus (VPL) of the thalamus. (B) Schematic representation of the structure of the cuneate layer.

In general, this study makes the following contributions:

1) To solve the problem of traditional RNNs being difficult to train for ultralong-sequence tasks, this paper proposes the CRNN inspired by the principle of mammalian cuneate circuit organization for filtering interference signals. The proposed CRNN employs a cuneate layer for signal extraction in the stacked recurrent neural layers, which effectively shortens the information propagation chain and prevents memory decline. The structure of the CRNN can be effectively trained in extremely long-sequence tasks, and at the same complexity as RNN, it is much longer than the sequence lengths that can be effectively analyzed by conventional RNN variants, and it has a faster optimization speed.
2) To demonstrate the feasibility of this research innovation, this paper provides theoretical arguments from the perspectives of gradient and SNR analysis. In gradient analysis, we propose a novel gradient analysis tool that can perform decomposition operations based on the chain rule of gradient calculation to simplify the complexity of gradient theory analysis.
3) To illustrate the effectiveness of the proposed CRNN, several experiments on ultralong-sequence tasks are designed in this paper to compare the training process and results of this network with several well-known sequence analysis models. The experimental results show that the CRNN achieves faster optimization and better fitting ability than other models.

2. Related work

In recent years, sequence analysis algorithms based on deep learning have been explored in different ways [31]. Preventing memory decline in neural networks and establishing long-term dependencies has been a long-standing research focus in the field of neural networks, and many solutions have been explored. In this section, we present advanced methods for improving the long-term memory capacity of neural networks.

2.1. Improved model based on a recurrent neural network

The most common current schemes incorporate information selection mechanisms in the neurons of RNN [32], such as long short-term memory (LSTM) and gated recurrent unit (GRU). To extend existing RNN models and shorten the effective size of the computational graph, Campos et al. used binary state update gates to learn to skip state updates [33]. This assisted in learning long-term dependencies. This improvement in the information selection pass mechanism increases the number of operations and parameters in the neural network, which can negatively affect computational efficiency. In addition, this improvement can control the SNR to some extent, and when the sequence length increases further, it still leads to memory decline due to the exponential decay effect of the effective information [34]. Several studies have suggested that the long-term memory of the RNNs may be enhanced by utilizing orthogonality-constrained model parameters [35, 36]. Further, Wisdom et al. proposed the optimization of unitary matrices along the Stiefel manifold using geodesic gradient descent [35]. However, some studies have shown [37] that this hard orthogonality constraint can impair the representational power of the model and may destabilize optimization. Several studies have investigated ways to alleviate the problem of memory decline by introducing additional skip connections across time or layers. Abbasimehr et al. proposed a prediction method based on multilayer LSTM networks applied to predict the actual demand data of furniture companies [38]. It can better capture nonlinear patterns in time series data while taking into account the inherent characteristics of nonstationary time series data. It has been proposed to create shortcut paths from the lower layers for deep networks with multiple LSTM layers to avoid conflicts between spatial and temporal-domain gradient flows [39]. Several studies have improved the architecture of multilayer RNNs based on the connection mechanism [25, 40] to better maintain the gradient magnitude in the vertical direction. These current architectures have more limited performance improvements for neural networks; skip connections across layers cannot shorten the path of gradient propagation along the temporal sequence, while skip connections across time break the continuity of the sequence, reduce the recognition accuracy of the neural network, and cannot adapt to dynamic length sequence inputs. Due to the inclusion of additional connections and computations, these structures increase model complexity and negatively affect optimization efficiency.

Although various improvements have been proposed for memory recession of RNN, the existing improvements currently suffer from complex structure, high operation, and low recognition accuracy, which is not ideal and hardly applicable to longer tasks. In this regard, this study proposes a novel, effective, and practical network structure as a solution to solve the problem that the above methods are not effective in dealing with ultralong-sequence tasks.

2.2. Research on sequence processing based on convolutional neural network

The important challenge of RNNs in dealing with long-sequence problems is the difficulty of establishing long-term dependencies and removing signal interference. In response, alternative or partial alternatives to RNN using convolutional neural networks (CNN) have been proposed, such as CNN-LSTM [41], WaveNet [42] and temporal convolutional network (TCN) [43]. Such structures use a convolutional structure to sample long sequences to shorten the sequence length. It cleverly circumvents the problem of memory degradation of RNN in processing long-sequence data and increases the computational speed by parallel operations. However, the CNN-based model may lose the position information in the sequences [44]. The shallow receptive field of convolutional layer is small due to the limited size of the convolutional kernel. Therefore, deep-layer neural networks must be built and undergo multiple downsampling to improve the receptive field. Since a large amount of underlying information has been forgotten by the neural network after multiple downsampling, it is difficult for CNN to achieve long-range underlying relationship modeling; therefore, the accuracy of CNN-based models is less desirable in dealing with long-sequence problems [45].

2.3. Transformer

Transformer [30] is a popular model for sequence processing, which shortens the path length through a self-attention mechanism, giving it the same path length as the fully connected layer. Therefore, it has a global receptive field and enables the modeling of long-term dependencies. Due to the parallel operation of the self-attention mechanism in the transformer, it has a faster operation speed than the RNN. However, since self-attention is computed between all

points in the sequence, the complexity grows by a factor of the square of the sequence length. The complicated structure of transformer makes the structural design of the transformer limited by the user's computational resource conditions, which is not very realistic for low-cost sequence processing requirements [46]. Difficulties also exist in the sequence modeling of the transformer: it cannot model periodic regular languages or basic recursion [47]. In addition, a large number of studies have demonstrated that the transformer cannot obtain positions from relative position encoding, which has a significant negative impact on the transformer's fitting ability [48], and this deficiency is even more pronounced when dealing with long-sequence tasks.

Some studies have attempted to reduce the complexity of the Transformer's attention mechanism by compressing the input information into a lower-dimensional matrix, such as Linformer [49] and Linear Transformer [50]. These studies perform information mapping based on this matrix, which lowers the computational cost. However, this compression process may cause information loss or limit the attention mechanism to local information, preventing the neural network from obtaining global and complete information. Consequently, these studies often have lower fitting ability [51].

3. Background and problem statement

In this section, we discuss the structure of RNN and other sequence processing models and the information propagation paths in them, introducing the principles of their memory recession in terms of gradient and SNR.

In this paper, we analyze memory decline in neural networks based on gradients. We quantitatively represent the magnitude of the gradient by the average Frobenius norm of the gradient matrix. Specifically, the scoring function $\mathcal{F}$ is defined for the gradient matrix G:

$$\mathcal{F}(G) = \frac{\|G\|_F^2}{Height_G}. \tag{1}$$

Where $Height_G$ represents the height of matrix G. For the linear calculation $Y = WX$, when the mean of each of these matrices converges to zero, the score is consistent with:

$$E(\mathcal{F}(Y)) = E(\mathcal{F}(W))E(\mathcal{F}(X)). \tag{2}$$

With the scoring function $\mathcal{F}$, the mathematical expectation of the overall gradient score can be quickly computed from the local gradient. It should be noted that when the decomposed local gradient X is used for the dot product operation, the scoring function is computed as:

$$\mathcal{F}(G) = \frac{\|X\|_F^2}{Width_X \times Height_X}. \tag{3}$$

Where $Width_X$ represents the width of matrix X.

In this paper, the degree to which the neural network is subject to signal interference is analyzed based on the signal energy SNR. For the operation $M = S + N$, the SNR in M is computed as:

$$SNR(S|M) = 10\lg\frac{Var(S)}{Var(N)}. \tag{4}$$

3.1. Recurrent neural network

The RNN is a kind of neural network commonly used for sequential operations, which can temporarily store the previous input and make judgments based on the subsequent input. Due to its recursive computing property, this neural network achieves parameter sharing in the time dimension and exhibits better robustness and fitting ability than conventional fully connected neural networks. Where the calculation process of RNN at moment i is as follows:

$$h_i = f\left(W_x x_i + W_h h_{i-1} + b\right). \tag{5}$$

Now that we have started to analyze the gradient relation in RNN, we compute the gradient of the hidden state of h_i for h_{i+1} in the i-th moment:

$$\frac{\partial h_i}{\partial h_{i-1}} = f'\left(W_x x_i + W_h h_{i-1} + b\right) \odot W_h, \tag{6}$$

Therefore,

$$\frac{\partial h_t}{\partial h_i} = \prod_{i+1}^{t} \left(\frac{\partial h_j}{\partial h_{j-1}} \right). \tag{7}$$

Then the gradient of the RNN prediction h_t for the i-th input x_i of X is:

$$\frac{\partial h_t}{\partial x_i} = \frac{\partial h_t}{\partial h_i} \frac{\partial h_i}{\partial x_i} = \frac{\partial h_t}{\partial h_i} \left(f' \left(W_x x_i + W_h h_{i-1} + b \right) \odot W_x \right). \tag{8}$$

Calculation of gradient scores:

$$E\left(\mathcal{F}\left(\frac{\partial h_t}{\partial x_i} \right) \right) = E\left(\mathcal{F}\left(\frac{\partial h_t}{\partial h_i} \right) \mathcal{F}\left(\frac{\partial h_t}{\partial x_i} \right) \right). \tag{9}$$

The activation function f used in RNN is generally rectified linear unit (ReLU) or Sigmoid, and its derivative is necessarily not greater than 1. Therefore, for any input vector X:

$$\mathcal{F}\left(f'(X) \right) = \frac{1}{n} \sum_{1}^{n} f'\left(X_i \right)^2 \leq 1. \tag{10}$$

To keep the hidden states in the RNN as well as the input data distribution consistent and prevent RNN gradient explosion, W_x and W_h should satisfy:

$$\mathcal{F}\left(W_x \right) + \mathcal{F}\left(W_h \right) \leq 1. \tag{11}$$

Since $\mathcal{F}$ is a non-negative function, it follows that:

$$\mathcal{F}\left(W_x \right) \leq 1 \ \ and \ \ \mathcal{F}\left(W_h \right) \leq 1. \tag{12}$$

Therefore,

$$\begin{aligned} E\left(\mathcal{F}\left(\frac{\partial h_i}{\partial h_{i-1}} \right) \right) =& E(\mathcal{F}\left(f'\left(W_x x_i + W_h h_{i-1} + b \right) \right) \\ & \mathcal{F}\left(W_h \right)) < 1 \end{aligned} \tag{13}$$

$$\begin{aligned} E\left(\mathcal{F}\left(\frac{\partial h_i}{\partial x_i} \right) \right) =& E(\mathcal{F}\left(f'\left(W_x x_i + W_h h_{i-1} + b \right) \right) \\ & \mathcal{F}\left(W_x \right)) < 1. \end{aligned} \tag{14}$$

Analyze the gradient of h_t against x_i when $t \gg i$,

$$E\left(\mathcal{F}\left(\frac{\partial h_t}{\partial h_i} \right) \right) = E\left(\prod_{i+1}^{t} \mathcal{F}\left(\frac{\partial h_j}{\partial h_{j-1}} \right) \right) \approx 0 \tag{15}$$

$$E\left(\mathcal{F}\left(\frac{\partial h_t}{\partial x_i} \right) \right) = E\left(\mathcal{F}\left(\frac{\partial h_t}{\partial h_i} \right) \mathcal{F}\left(\frac{\partial h_i}{\partial x_i} \right) \right) \approx 0. \tag{16}$$

That is to say, the prediction results of RNN are not dependent on the input x_i which is temporally distant, and the memory of RNN for x_i can be considered to have declined.

To analyze the case of interference by signals in a standard RNN, the SNR of h_i for h_{i+1} in the hidden state is computed as

$$SNR\left(h_i \middle| h_{i+1} \right) = 10 \lg \frac{\mathcal{S}|_{h_i}^{h_{i+1}}}{\mathcal{N}|_{h_i}^{h_{i+1}}}, \tag{17}$$

where $\mathcal{S}|_{h_i}^{h_{i+1}}$ denotes the energy contributed by h_i to h_{i+1} and $\mathcal{N}|_{h_i}^{h_{i+1}}$ denotes the energy of the neural network subjected to noise during the process from state h_i to h_{i+1}. In Eq. (17) we consider h_i to be valid information, then the information added in the process of updating it to h_{i+1} can be considered as noise. Therefore:

$$\mathcal{S}|_{h_i}^{h_{i+1}} = Var\left(W_h h_i \right) \tag{18}$$

$$\mathcal{N}|_{h_i}^{h_{i+1}} = Var\left(W_x x_{i+1}\right). \tag{19}$$

Then the SNR of the predicted value h_t of the RNN for its a-th hidden state h_a is:

$$SNR\left(h_a|h_t\right) = 10\lg\frac{\mathcal{S}|_{h_a}^{h_t}}{\mathcal{N}|_{h_a}^{h_t}} = 10\lg\frac{\mathcal{S}|_{h_a}^{h_{a+1}}\Theta|_{h_{a+1}}^{h_t}}{\mathcal{N}|_{h_a}^{h_t}}, \tag{20}$$

where $\Theta|_{h_{a+1}}^{h_t}$ represents the proportion of energy in the state h_{a+1} that decays by h_t, in Eq. (20):

$$\Theta|_{h_{a+1}}^{h_t} = \prod_{a+1}^{t-1}\frac{Var\left(W_h h_i\right)}{Var\left(f^{-1}\left(h_i\right)\right)} \tag{21}$$

$$\begin{aligned}\mathcal{N}|_{h_a}^{h_t} &= \sum_{a+1}^{t}\left(\mathcal{N}|_{h_{i-1}}^{h_i}\Theta|_{h_i}^{h_t}\right)\\ &= \sum_{a+1}^{t}\left(Var\left(W_x x_i\right)\prod_{i}^{t-1}\frac{Var\left(W_h h_j\right)}{Var\left(f^{-1}\left(h_j\right)\right)}\right).\end{aligned} \tag{22}$$

Set for arbitrary h and x:

$$E\left(\frac{Var\left(W_h h\right)}{Var\left(f^{-1}\left(h\right)\right)}\right) = \varphi \tag{23}$$

$$E\left(\frac{Var\left(W_h h\right)}{Var\left(W_x x\right)}\right) = \lambda. \tag{24}$$

Then,

$$E\left(SNR\left(h_a|h_t\right)\right) = 10\lg\frac{\varphi^{t-a-1}}{\left(\frac{1-\varphi^{t-a-1}}{1-\varphi}\right)} + 10\lg\lambda. \tag{25}$$

Since:

$$E\left(Var\left(f^{-1}\left(h_{i+1}\right)\right)\right) = E\left(Var\left(W_h h_i\right) + Var\left(W_x x_{i+1}\right)\right). \tag{26}$$

Therefore,

$$\varphi = E\left(\frac{Var\left(W_h h\right)}{Var\left(f^{-1}\left(h\right)\right)}\right) \leq 1. \tag{27}$$

When $t \gg a$:

$$E\left(SNR\left(h_a|h_t\right)\right) \approx 10\left(t-a-2\right)\lg\varphi + 10\lg\lambda\left(1-\varphi\right). \tag{28}$$

The SNR at this point is a very low negative value and is negatively correlated with t-a, indicating that at this point h_t contains essentially no energy about the temporally distant h_a.

From the above derivation, it can be concluded that RNNs can only retain short-term memory and cannot work properly when the processed sequences are longer. This is because RNNs are highly influenced by interference signals when computing long-sequence problems for neurons. In response, researchers have proposed many remedies, such as LSTM, GRU, Residual LSTM, and other structures.

3.2. LSTM

LSTM is one of the most common RNN variants, and its most important feature is the addition of memory cells for storing long-term memory in its neurons and controlling the retention and forgetting of information through multiple gating functions. This structure can greatly slow down the rate of information forgetting and reduce the SNR. The calculation of LSTM is as follows:

$$\begin{vmatrix} f_i \\ g_i \\ o_i \end{vmatrix} = \sigma\left(\begin{vmatrix} W_{fh} & W_{fx} & b_f \\ W_{gh} & W_{gx} & b_g \\ W_{oh} & W_{ox} & b_o \end{vmatrix}\begin{vmatrix} h_{i-1} \\ x_i \\ 1 \end{vmatrix}\right) \tag{29}$$

$$\hat{c}_i = tanh\left(W_{ch}h_{i-1} + W_{cx}x_i + b_c\right) \tag{30}$$

$$c_i = f_i \odot c_{i-1} + g_i \odot \hat{c}_i \tag{31}$$

$$h_i = o_i \odot tanh\left(c_i\right), \tag{32}$$

where c is a memory cell, and since memory cells are responsible for storing long-term memory in the LSTM, memory cells are used here as the object of gradient and SNR derivation.

We compute the gradient of the memory cells of c_i for c_{i-1}:

$$\frac{\partial c_i}{\partial c_{i-1}} = f_i. \tag{33}$$

Therefore,

$$\frac{\partial c_t}{\partial c_i} = \prod_{i+1}^{t} f_j. \tag{34}$$

Then the gradient of the predicted value h_t of the LSTM for x_i is:

$$\frac{\partial h_t}{\partial x_i} = \frac{\partial h_t}{\partial c_t}\frac{\partial c_t}{\partial c_i}\frac{\partial c_i}{\partial x_i}. \tag{35}$$

In Eq. (35):

$$\frac{\partial h_t}{\partial c_t} = o_t \odot tanh'\left(c_t\right) \tag{36}$$

$$\begin{aligned}\frac{\partial c_i}{\partial x_i} &= \hat{c}_i \odot \sigma'\left(\sigma^{-1}\left(g_i\right)\right) \odot W_{gx} \\ &\quad + g_i \odot tanh'\left(tanh^{-1}\left(\hat{c}_i\right)\right) \odot W_{cx} \\ &= \hat{c}_i \odot g_i \odot \left(1 - g_i\right) \odot W_{gx} + g_i \odot \left(1 - \hat{c}_i^2\right) \odot W_{cx}.\end{aligned} \tag{37}$$

The gradient amplitude score is:

$$E\left(\mathcal{F}\left(\frac{\partial h_t}{\partial x_i}\right)\right) = E\left(\mathcal{F}\left(\frac{\partial h_t}{\partial c_t}\right)\mathcal{F}\left(\frac{\partial c_t}{\partial c_i}\right)\mathcal{F}\left(\frac{\partial c_i}{\partial x_i}\right)\right). \tag{38}$$

Since for any input X and activation function f:

$$\mathcal{F}\left(f\left(X\right)\right) \leq 1 \tag{39}$$

$$\mathcal{F}\left(f'\left(X\right)\right) \leq 1. \tag{40}$$

Therefore,

$$\mathcal{F}\left(\frac{\partial h_t}{\partial c_t}\right) = \mathcal{F}\left(o_t \odot tanh'\left(c_t\right)\right) \leq 1 \tag{41}$$

$$E\left(\mathcal{F}\left(\frac{\partial c_t}{\partial c_i}\right)\right) = E\left(\prod_{i+1}^{t}\mathcal{F}\left(f_j\right)\right) \leq 1 \tag{42}$$

$$\begin{aligned}E\left(\mathcal{F}\left(\frac{\partial c_i}{\partial x_i}\right)\right) &\leq E(\mathcal{F}(\hat{c}_i \odot g_i \odot \left(1 - g_i\right) \\ &\quad + g_i \odot \left(1 - \hat{c}_i^2\right))) \leq 1.\end{aligned} \tag{43}$$

Analyze the gradient of h_t against x_i when $t \gg i$:

$$E\left(\mathcal{F}\left(\frac{\partial c_t}{\partial c_i}\right)\right) = E\left(\prod_{i+1}^{t}\mathcal{F}\left(f_j\right)\right) \approx 0 \tag{44}$$

$$E\left(\mathcal{F}\left(\frac{\partial h_t}{\partial x_i}\right)\right) = E\left(\mathcal{F}(\frac{\partial h_t}{\partial c_t})\mathcal{F}(\frac{\partial c_t}{\partial c_i})\mathcal{F}(\frac{\partial c_i}{\partial x_i})\right) \approx 0. \tag{45}$$

This indicates that the prediction results of LSTM also largely do not depend on the temporally distant input x_i, suggesting that although LSTM can alleviate memory decline to some extent, it cannot completely solve the problem.

To analyze the situation of being subject to signal interference in LSTM, we compute the SNR of c_i to c_{i+1}:

$$SNR\left(c_i|c_{i+1}\right) = 10\lg\frac{\mathcal{S}|_{h_i}^{h_{i+1}}}{\mathcal{N}|_{h_i}^{h_{i+1}}}. \tag{46}$$

In Eq. (46):

$$\mathcal{S}|_{h_i}^{h_{i+1}} = Var\left(f_{i+1}\odot c_i\right) \tag{47}$$

$$\mathcal{N}|_{h_i}^{h_{i+1}} = Var\left(g_{i+1}\odot \hat{c}_{i+1}\right). \tag{48}$$

Then the SNR of the final memory c_t of the LSTM for its a-th memory cell c_a is

$$SNR\left(c_a|c_t\right) = 10\lg\frac{\mathcal{S}|_{c_a}^{c_t}}{\mathcal{N}|_{c_a}^{c_t}} = 10\lg\frac{\mathcal{S}|_{c_a}^{c_{a+1}}\Theta|_{c_{a+1}}^{c_t}}{\mathcal{N}|_{c_a}^{c_t}}. \tag{49}$$

In Eq. (49):

$$\Theta|_{c_{a+1}}^{c_t} = \prod_{a+1}^{t-1}\mathcal{F}\left(f_{i+1}\right) \tag{50}$$

$$\begin{aligned}\mathcal{N}|_{c_a}^{c_t} &= \sum_{a+1}^{t}\left(\mathcal{N}|_{c_{i-1}}^{c_i}\Theta|_{c_i}^{c_t}\right)\\ &= \sum_{a+1}^{t}\left(Var\left(g_i\odot\hat{c}_i\right)\prod_{i}^{t-1}\mathcal{F}\left(f_{j+1}\right)\right).\end{aligned} \tag{51}$$

Set for arbitrary f and c:

$$E\left(S\left(f\right)\right) = \varphi \tag{52}$$

$$E\left(\frac{Var\left(f\odot c\right)}{Var\left(g\odot\hat{c}\right)}\right) = \lambda. \tag{53}$$

Obviously,

$$\varphi \le 1 \tag{54}$$

$$E\left(SNR\left(c_a|c_t\right)\right) = 10\lg\frac{\varphi^{t-a-1}}{\left(\frac{1-\varphi^{t-a-1}}{1-\varphi}\right)} + 10\lg\lambda. \tag{55}$$

This equation is the same as the SNR formulation of RNN and therefore leads to the same conclusion: when $t \gg a$, the SNR is a very small value, indicating that the LSTM cannot completely solve the memory decline problem.

3.3. Other improved models based on RNN

Researchers have proposed many other RNN improvement schemes, such as multilayer LSTM, residual LSTM, and bidirectional RNN (BIRNN). However, these network structures are essentially just improving the memory selection pass mechanism or increasing the number of layers and connections of the neural network. They cannot completely solve the problems of signal interference and memory decline in ultralong sequences [34], and the theoretical arguments of the above conclusions are omitted due to space limitations.

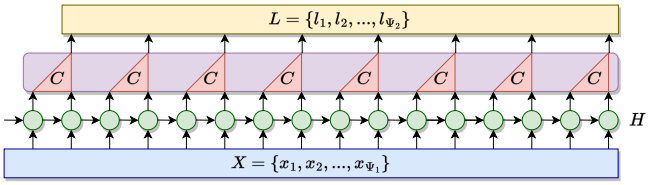

Fig. 2. A C-Block consists of an RNN and a cuneate layer.

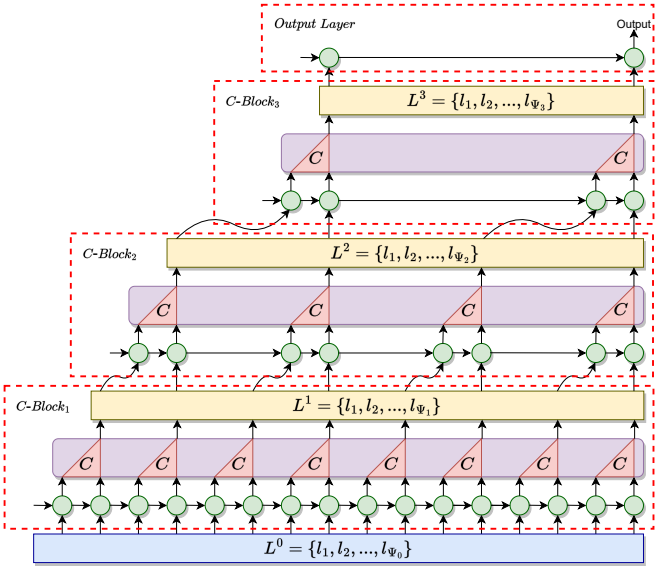

Fig. 3. A simple cuneate RNN structure, the network in the figure has 3 C-Blocks and the cuneate layer sampling period is 2.

4. Cuneate recurrent neural network

In this section, we will describe in detail the CRNN proposed in this study, demonstrate its effectiveness in preventing memory decline and improving the SNR, and then compare it to the structural metrics of several popular current sequence processing models.

4.1. Network structure

A CRNN is a stacked structure consisting of RNNs and cuneate layers, where the output tensor of each RNN layer is fed into the next cuneate layer. The cuneate layer de-noise samples the output sequence of the RNN with period T and feeds it into the next RNN. Here, an RNN and a cuneate layer are combined as a Cuneate-Block (C-Block), and the input sequence X of the C-Block is of length Ψ_1 and the output sequence L is of length Ψ_2. Then the relationship between Ψ_1 and Ψ_2 is:

$$\Psi_1 = T\Psi_2. \tag{56}$$

The structure of a C-Block is shown in Fig. 2, and its complete operation is shown as:

$$X = \left[x_1, x_2, \cdots, x_{\Psi_1}\right] \tag{57}$$

$$h_i = f\left(W_x x_i + W_h h_{i-1} + b\right) \tag{58}$$

$$l_i = C\left(h_{T(i-1)+1}, \cdots, h_{T(i-1)+T}\right) \tag{59}$$

$$L = \left[l_1, l_2, \cdots, l_{\Psi_2}\right], \tag{60}$$

where C denotes the denoising sampling function in the cuneate layer, the exact form of which can be defined according to the user's needs and will be discussed in detail in Section 4.3.

A complete CRNN consists of n layers of C-Blocks, the input tensor of the input layer is L^0, and the output of each C-Block in the post-order is L^1, L^2, $L^3\cdots$. The output layer of the CRNN is an RNN with L^n as the input, and the RNN analyzes the high-level signal sampled by multiple layers of denoising to obtain the final prediction. The specific structure of the CRNN is shown in Fig. 3. The sampling period in the illustration is 2, and the user can change the sampling period according to demand. We suggest that the value of the sampling period be set in the range of 2–16.

4.2. Theoretical arguments

To demonstrate the effectiveness of cuneate recurrent networks in improving the SNR and preventing memory decline, this section analyzes the gradient and SNR of cuneate recurrent networks.

First, we perform the gradient analysis, and the following gradient relations are computed in a single C-Block:

$$\frac{\partial h_j}{\partial h_i} = \begin{cases} \prod_i^j \left(\frac{\partial h_{k+1}}{\partial h_k} \right), & j > i \\ 1, & j = i \\ 0, & j < i \end{cases} \tag{61}$$

$$dH = \begin{bmatrix} \frac{\partial h_1}{\partial h_1} & \cdots & \frac{\partial h_1}{\partial h_{S_1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_{\Psi_1}}{\partial h_1} & \cdots & \frac{\partial h_{\Psi_1}}{\partial h_{\Psi_1}} \end{bmatrix} \tag{62}$$

$$\frac{\partial L}{\partial H} = C'(H)\, dH \tag{63}$$

$$\frac{\partial H}{\partial X} = f'\left(W_x X + W_h H + b\right) \odot W_x. \tag{64}$$

Then the gradient of the output L to the input X is:

$$\frac{\partial L}{\partial X} = \frac{\partial L}{\partial H}\frac{\partial H}{\partial X} = C'(H)\, dH \left(f'\left(W_x X + W_h H + b\right) \odot W_x\right). \tag{65}$$

The gradient of the hidden state matrix H of the cuneate network output to the output layer is:

$$\frac{\partial Output}{\partial H} = \left[\frac{\partial h_t}{\partial h_1}, \frac{\partial h_t}{\partial h_2}, \cdots, \frac{\partial h_t}{\partial h_t}\right]. \tag{66}$$

The gradient of $Output$ to the last C-Block output L^n is:

$$\begin{aligned} \frac{\partial Output}{\partial L^n} &= \frac{\partial Output}{\partial H}\frac{\partial H}{\partial L^n} \\ &= \frac{\partial Output}{\partial H}\left(f'\left(W_x X + W_h H + b\right) \odot W_x\right). \end{aligned} \tag{67}$$

The gradient of L^n for the input layer L^0 is:

$$\begin{aligned} \frac{\partial L^n}{\partial L^0} = \prod_1^n (&(C^{i\prime}\left(L^i\right) dH^i) \\ &\left(f'\left(W_x^i L^{i-1} + W_h^i H^i + b^i\right) \odot W_x^i\right)). \end{aligned} \tag{68}$$

The gradient of $Output$ for the input layer is:

$$\frac{\partial Output}{\partial L^0} = \frac{\partial Output}{\partial L^n}\frac{\partial L^n}{\partial L^0}. \tag{69}$$

Let the sequence length of L^n be t. The relationship between t and the sequence length Ψ_0 of the input layer is:

$$\Psi_0 = tT^n. \tag{70}$$

Therefore, $t \ll \Psi_0$. The value of t is small and there is no $t \gg i$. Therefore, for any $\frac{\partial h_t}{\partial h_i}$, $\mathcal{F}\left(\frac{\partial h_t}{\partial h_i}\right)$ has a small effect on the overall gradient magnitude score and can be considered as:

$$0 \leqq \mathcal{F}\left(\frac{\partial Output}{\partial L^n}\right) < 1. \tag{71}$$

The gradient amplitude scores of $\frac{\partial L^n}{\partial L^0}$ are computed as:

$$\begin{aligned} E\left(\mathcal{F}\left(\frac{\partial L^n}{\partial L^0}\right)\right) = \prod_1^n E(&\mathcal{F}\left(C^{i\prime}\left(L^i\right)\right)\mathcal{F}\left(dH^i\right) \\ &\mathcal{F}\left(f'\left(W_x^i L^{i-1} + W_h^i H^i + b^i\right)\right)\mathcal{F}\left(W_x^i\right)), \end{aligned} \tag{72}$$

where the score $\mathcal{F}\left(\frac{\partial L^n}{\partial L^0}\right)$ is determined by the user-defined sampling function C^i, which can be considered as $\mathcal{F}\left(C^{i\prime}\left(L^i\right)\right) \approx 1$. For any dH, it satisfies:

$$\mathcal{F}(dH) = \frac{\sum_1^{Height_{dH}} \sum_1^{Width_{dH}} \left(\frac{\partial h_j}{\partial h_i}\right)^2}{Height_{dH}} \geq 1. \tag{73}$$

Based on the conclusion of Section 3.1:

$$0 \leqq \mathcal{F}\left(f'\left(W_x^i L^{i-1} + W_h^i H^i + b^i\right)\right)\mathcal{F}\left(W_x^i\right) < 1. \tag{74}$$

Since n is the number of network layers, which is a small natural number, for any C-Block:

$$0 \leqq E\left(\mathcal{F}\left(\frac{\partial L^n}{\partial L^0}\right)\right) < 1. \tag{75}$$

Therefore,

$$0 \leqq E\left(\mathcal{F}\left(\frac{\partial Output}{\partial L^0}\right)\right) < 1. \tag{76}$$

This shows that CRNN has a large gradient for all of its input information, and it can better remember the needed information without memory decline.

Next, we compute the SNR of the CRNN, and the SNR of the RNN input value x_i for h_i is:

$$SNR\left(x_i \middle| h_i\right) = 10 \lg \frac{\mathcal{S}|_{x_i}^{h_i}}{\mathcal{N}|_{x_i}^{h_i}}. \tag{77}$$

In Eq. (77):

$$\mathcal{S}|_{x_i}^{h_i} = Var\left(W_x x_i\right) \tag{78}$$

$$\mathcal{N}|_{x_i}^{h_i} = Var\left(W_h h_{i-1}\right). \tag{79}$$

The sampling function C has the function of filtering noise, so in the ideal case, the SNR of its output value is the same as its input with the highest SNR. Therefore, in the j-th sampling period, its output value l_j satisfies:

$$SNR\left(x_i \middle| l_j\right) = Max(SNR\left(x_i \middle| h_{Tj+1}\right), SNR\left(x_i \middle| h_{Tj+2}\right),$$

$$\cdots, SNR\left(x_i \middle| h_{Tj+T}\right)). \tag{80}$$

When $i \in [Tj+1, Tj+T]$ in Eq. (80), i.e., h_i is included in the j-th sampling period:

$$SNR\left(x_i \middle| l_j\right) = SNR\left(x_i | h_i\right). \tag{81}$$

It can be expressed as:

$$SNR\left(x_i \middle| l_{i|T}\right) = SNR\left(x_i | h_i\right). \tag{82}$$

When $i \notin [Tj+1, Tj+T]$ in Eq. (80), the SNR of l_j is low and can be neglected because the RNN is subject to large signal interference. To simplify the operation, the following analysis computes only the SNR of the output that is in the same sampling period as the input signal.

It can be computed that the SNR of the $a|T^n$-th hidden state $H_{a|T^n}^{n+1}$ in the output layer to the a-th input signal L_a^0 in the input layer is:

$$\begin{aligned} SNR\left(L_a^0 \middle| H_{a|T^n}^{n+1}\right) &= 10\lg \frac{\mathcal{S}\left|\begin{matrix} H_{a|T^n}^{n+1} \\ L_a^0 \end{matrix}\right.}{\mathcal{N}\left|\begin{matrix} H_{a|T^n}^{n+1} \\ L_a^0 \end{matrix}\right.} \\ &= 10\lg \frac{\mathcal{S}\left|\begin{matrix} H_a^1 \\ L_a^0 \end{matrix}\right. \Theta \left|\begin{matrix} H_{a|T^n}^{n+1} \\ H_a^1 \end{matrix}\right.}{\mathcal{N}\left|\begin{matrix} H_{a|T^n}^{n+1} \\ L_a^0 \end{matrix}\right.} \\ &= 10\lg \frac{\mathcal{S}\left|\begin{matrix} H_a^1 \\ L_a^0 \end{matrix}\right. \Theta \left|\begin{matrix} H_{a|T^n}^{n+1} \\ L_{a|T}^1 \end{matrix}\right.}{\mathcal{N}\left|\begin{matrix} H_{a|T^n}^{n+1} \\ L_a^0 \end{matrix}\right.}. \end{aligned} \tag{83}$$

In Eq. (83):

$$\mathcal{S}\left|\begin{matrix} H_a^1 \\ L_a^0 \end{matrix}\right. = Var\left(W_x^1 L_a^0\right) \tag{84}$$

$$\Theta\left|\begin{matrix} H_{a|T^n}^{n+1} \\ L_{a|T}^1 \end{matrix}\right. = \prod_1^n \frac{Var\left(W_x^{i+1} L_{a|T^i}^i\right)}{Var\left(f^{-1}\left(H_{a|T^{i-1}}^i\right)\right)} \tag{85}$$

$$\begin{aligned} \mathcal{N}\left|\begin{matrix} H_{a|T^n}^{n+1} \\ L_a^0 \end{matrix}\right. &= \sum_1^{n+1}\left(\mathcal{N}\left|\begin{matrix} L_{a|T^i}^i \\ L_{a|T^{i-1}}^{i-1} \end{matrix}\right. \Theta \left|\begin{matrix} H_{a|T^n}^{n+1} \\ L_{a|T^i}^i \end{matrix}\right.\right) \\ &= \sum_1^{n+1}(Var\left(W_h^i H_{a|T^{i-1}-1}^i\right) \\ &\quad \prod_i^n \frac{Var\left(W_x^{j+1} L_{a|T^j}^i\right)}{Var\left(f^{-1}\left(H_{a|T^{j-1}}^i\right)\right)}). \end{aligned} \tag{86}$$

In the output layer, the SNR of the predicted value H_t^{n+1} for L_a^0 is:

$$SNR\left(L_a^0 \middle| H_t^{n+1}\right) = 10\lg \frac{\mathcal{S}\left|\begin{matrix} H_t^{n+1} \\ L_a^0 \end{matrix}\right.}{\mathcal{N}\left|\begin{matrix} H_t^{n+1} \\ L_a^0 \end{matrix}\right.} \tag{87}$$

$$= 10\lg \frac{\mathcal{S}\left|\begin{matrix} H_{a|T^n}^{n+1} \\ L_a^0 \end{matrix}\right. \Theta \left|\begin{matrix} H_t^{n+1} \\ H_{a|T^n}^{n+1} \end{matrix}\right.}{\mathcal{N}\left|\begin{matrix} H_{a|T^n}^{n+1} \\ L_a^0 \end{matrix}\right. \Theta \left|\begin{matrix} H_t^{n+1} \\ H_{a|T^n}^{n+1} \end{matrix}\right. + \mathcal{N}\left|\begin{matrix} H_t^{n+1} \\ H_{a|T^n}^{n+1} \end{matrix}\right.}.$$

In Eq. (87):

$$\Theta\left|\begin{matrix} H_t^{n+1} \\ H_{a|T^n}^{n+1} \end{matrix}\right. = \prod_{a|T^n}^{t-1} \frac{Var\left(W_h^{n+1} H_i^{n+1}\right)}{Var\left(f^{-1}\left(H_i^{n+1}\right)\right)} \tag{88}$$

$$\begin{aligned} \mathcal{N}\left|\begin{matrix} H_t^{n+1} \\ H_{a|T^n}^{n+1} \end{matrix}\right. &= \sum_{a|T^n+1}^{t} \left(\mathcal{N}\left|\begin{matrix} H_i^{n+1} \\ H_{i-1}^{n+1} \end{matrix}\right. \Theta \left|\begin{matrix} H_t^{n+1} \\ H_i^{n+1} \end{matrix}\right.\right) \\ &= \sum_{a|T^n+1}^{t} \left(Var\left(W_x^{n+1} L_i^n\right) \prod_{i}^{t-1} \frac{Var\left(W_h^{n+1} H_j^{n+1}\right)}{Var\left(f^{-1}\left(H_j^{n+1}\right)\right)}\right). \end{aligned} \tag{89}$$

Set for arbitrary H and L:

$$E\left(\frac{Var\left(W_h H\right)}{Var\left(f^{-1}\left(H\right)\right)}\right) = \varphi \tag{90}$$

$$E\left(\frac{Var\left(W_h H\right)}{Var\left(W_x L\right)}\right) = \lambda. \tag{91}$$

Therefore,

$$E\left(\Theta\left|\begin{matrix} H_{a|T^n}^{n+1} \\ L_{a|T}^1 \end{matrix}\right.\right) = \prod_{1}^{n} \frac{\varphi}{\lambda} \tag{92}$$

$$E\left(\frac{\mathcal{N}\left|\begin{matrix} H_{a|T^n}^{n+1} \\ L_a^0 \end{matrix}\right.}{\mathcal{S}\left|\begin{matrix} H_a^1 \\ L_a^0 \end{matrix}\right.}\right) = \sum_{1}^{n+1} \left(\frac{1}{\lambda} \prod_{i}^{n} \frac{\varphi}{\lambda}\right) \tag{93}$$

$$E\left(\Theta\left|\begin{matrix} H_t^{n+1} \\ H_{a|T^n}^{n+1} \end{matrix}\right.\right) = \prod_{a|T^n}^{t-1} \varphi \tag{94}$$

$$E\left(\frac{\mathcal{N}\left|\begin{matrix} H_t^{n+1} \\ H_{a|T^n}^{n+1} \end{matrix}\right.}{\mathcal{S}\left|\begin{matrix} H_a^1 \\ L_a^0 \end{matrix}\right.}\right) = \sum_{a|T^n+1}^{t} \left(\prod_{i}^{t-1} \varphi\right) \tag{95}$$

$$E\left(SNR\left(L_a^0 \middle| H_t^{n+1}\right)\right) = \tag{96}$$

$$\begin{aligned} &10\lg \frac{\prod_1^n \frac{\varphi}{\lambda} \prod_{a|T^n}^{t-1} \varphi}{\sum_1^{n+1} \left(\frac{1}{\lambda} \prod_i^n \frac{\varphi}{\lambda}\right) \prod_{a|T^n}^{t-1} \varphi + \sum_{a|T^n+1}^{t} \left(\prod_i^{t-1} \varphi\right)} \\ &= 10\lg \frac{\prod_1^n \frac{\varphi}{\lambda}}{\sum_1^{n+1} \left(\frac{1}{\lambda} \prod_i^n \frac{\varphi}{\lambda}\right) + \sum_{a|T^n+1}^{t} \left(\prod_{a|T^n}^{i} \varphi\right)} \end{aligned}$$

$$E\left(SNR\left(x_a|o_t\right)\right) = E\left(SNR\left(L_a^0\middle|H_t^{n+1}\right)\right)$$
$$= 10\lg\frac{\left(\frac{\varphi}{\lambda}\right)^n}{\frac{1-\left(\frac{\varphi}{\lambda}\right)^n}{\lambda\left(1-\frac{\varphi}{\lambda}\right)} + \frac{\varphi^2-\varphi^{(\Psi_0-a)|T^n+1}}{1-\varphi}}. \tag{97}$$

Since

$$E\left(Var\left(f^{-1}\left(H\right)\right)\right) = E\left(Var\left(W_hH\right) + Var\left(W_xL\right)\right), \tag{98}$$

it follows that:

$$\frac{\varphi}{\lambda} = E\left(\frac{Var\left(W_xL\right)}{Var\left(f^{-1}\left(H\right)\right)}\right) < 1. \tag{99}$$

Therefore,

$$E\left(SNR\left(x_a|o_t\right)\right) \geq 10\lg\frac{\left(\frac{\varphi}{\lambda}\right)^n}{\frac{n+1}{\lambda} + \left(\Psi_0 - a\right)|T^n} \tag{100}$$
$$= 10n\log_{10}\frac{\varphi}{\lambda} - 10\lg\left(\frac{n+1}{\lambda} + \left(\Psi_0 - a\right)|T^n\right).$$

Since both n and $\left(\Psi_0 - a\right)|T^n$ are smaller values, the SNR score of $E\left(SNR\left(x_a|o_t\right)\right)$ is higher, and its SNR decreases linearly with the increase of the number of C-Block layers n and is less affected by the position of x_a, which indicates that the CRNN suffers less signal interference in the operation.

From the above derivation, it can be seen that the gradient and SNR of CRNN can be maintained at a high level and are less affected by the length of the input sequence.

4.3. Cuneate sampling function

The cuneate layer is an important component of CRNN, which extracts important information from the input signal and filters information that is not relevant to the predicted behavior. The following are several optional cuneate sampling functions, which are each suitable for different types of tasks. Attention-based sampling functions:

$$l_i = Sum(Softmax\left(W_c\left[h_{T(i-1)+1}, \cdots, h_{T(i-1)+T}\right]\right)$$
$$\odot \left[h_{T(i-1)+1}, \cdots, h_{T(i-1)+T}\right]). \tag{101}$$

Periodic sampling function:

$$l_i = h_{T(i-1)+T}. \tag{102}$$

Linear sampling function:

$$l_i = W_cConcat\left(\left[h_{T(i-1)+1}, \cdots, h_{T(i-1)+T}\right]\right). \tag{103}$$

Slice sampling function:

$$l_i = h_{\Psi-T+i}. \tag{104}$$

We recommend using the attention-based sampling function, which achieves better information filtering and preserves location information. Unlike self-attention, this attention mechanism operates within a certain range, scoring the importance of the information in that region, and fusing the information based on the scores before feeding it to the next layer. This algorithm has the characteristics of low complexity and fast optimization. The periodic sampling function is simple to use and can be employed during model debugging. The linear sampling function, on the other hand, has the advantage of retaining relatively complete information and can be tried when the attention-based sampling function performs poorly. The slice sampling function retains only the end of the sequence in each sampling and is suitable for handling sequence prediction tasks (e.g., temperature prediction, language modeling, and music modeling tasks), but it performs poorly in regular pattern analysis tasks and is generally not used in principle.

4.4. Model design

The CRNN proposed in this study is a flexible and variable structure, and users can modify the internal structure by themselves according to usage requirements. In our experiments, we found that replacing RNNs with bidirectional RNNs can improve the fitting accuracy to a greater extent because bidirectional RNNs can circumvent the drawback of unidirectional data propagation of conventional RNNs. However, this paper does not recommend using LSTM and GRU instead of RNN, due to the limited optimization effect of such networks for ultralong-sequence tasks and the great slowdown of computing speed.

We recommend using unsaturated functions, such as ReLU and Leaky ReLU, as activation functions for this network. This is because such functions have a wider gradient sensitivity range, which can prevent gradient disappearance and improve optimization speed. To improve the stability of the gradient during training, we suggest performing a layer normalization operation on the output of each RNN layer.

4.5. Model analysis

One of the outstanding advantages of CRNN over other sequence processing models is its linear complexity, very small maximum path length, and maximum perceptual field. Table 1 compares the complexity, maximum path length, and perceptual field of the CRNN and several common neural network layers. It can be seen that CRNN is the best among the compared networks in terms of these metrics. Thanks to this, CRNN can build long-term dependencies in long sequences with very little cost.

Table 1
Comparison of Complexity, Maximum Path Length, and Perceptual Field of Various Neural Network Layers [52].

Layer Type	Complexity per Layer	Maximum Path Length	Receptive Field
Self-Attention	$O(L^2D)$	$O(1)$	$O(L)$
Fully Connected	$O(L^2D^2)$	$O(1)$	$O(L)$
Convolutional	$O(KLD^2)$	$O(log_K L)$	$O(K)$
Recurrent	$O(LD^2)$	$O(1)$	$O(L)$
LSTM	$O(8L^2D)$	$O(1)$	$O(L)$
CRNN	$O(LD^2)$	$O(log_T L)$	$O(L)$

“L” is the sequence length, “D” is the number of hidden state dimensions, “K” is the convolution kernel size, and “T” is the sampling period of the cuneate layer.

In terms of the computational structure of CRNN, it is more in line with the human process of understanding things. Its shallow C-Blocks are responsible for collecting signals and extracting low-level features. The subsequent stacked C-Blocks continuously refine and analyze the features, and the output layer makes judgments based on the refined abstract information. This enables the first few layers of the neural network to focus on feature extraction, while the deeper layers are used to process high-level features.

5. Experiments

To fully validate the effectiveness of the proposed network in this study, it was tested using several datasets and compared with several common sequence processing models. The datasets used include the GTZAN music genre dataset [53], a shuffled version of Mixed National Institute of Standards and Technology database (pMNIST) [54], the XJTU-SY bearing remaining useful life (RUL) dataset [49], the WOS-46985 document classification dataset [55], and the Penn Treebank (PTB) text dataset [56]. The common feature of these datasets is that they all have long signal sequence lengths, which are suitable for comparing the performance of different neural networks in long-sequence tasks. The neural networks involved in the experimental comparison are standard RNN, multilayer RNN, multilayer BIRNN, multilayer BIGRU, multilayer BILSTM, WaveNet, TCN, and CNN-LSTM. In the last layer of each neural network, we add a linear layer as the output layer. Due to the excessive space occupied by the transformer in computing long-sequence data, which exceeds the memory limit of the equipment used by our team, the transformer model is not included in this paper for comparison.

In the experimental design of CRNN, we all use the recommended metrics for model design in Section 4.4 as the basis. The bidirectional RNN is chosen as the recurrent network structure in which the activation function is set to ReLU, and layer normalization is performed on all BIRNN layer outputs. We set the corresponding cuneate sampling functions according to the situation of the different experimental tasks, which will be described in the corresponding subsection of each experiment.

Since the purpose of the experiments in this study is to compare the performance of each neural network in the long-sequence task, we generate longer sequence data as the input of the neural network during the data preprocessing. Only the best performance of different networks under the same complexity conditions is provided in the training results below to fully ensure the fairness of the comparison. In the experiments for each dataset, we recorded the relationship between its testing accuracy and training times and used it as the basis for the optimization speed comparison.

In our experiments, we all used Pytorch for neural network construction and Adam [57] as an optimizer, and trained on a single RTX 3090 graphics processing unit.

5.1. GTZAN music genre classification

GTZAN is a music dataset containing a total of 1000 segments of 30s music waveform data, which are classified into 10 music genres with a sampling frequency of 22.05 kHz. The goal of the neural network in this task is to accurately classify each music genre. We first preprocessed the waveform data with a short-term Fourier transform (STFT) of Hann window size 32 and divided each time-frequency map into five segments in time dimension, resulting in 5000 time-frequency maps with a time domain length of 8250. The 4000 segments are used as the training set and the 1000 segments are used as the validation set, and the data with a time domain length of 8192 is intercepted from each segment as the input of the neural network during training. We use the attention-based sampling function as the cuneate layer of the CRNN. The variation in classification accuracy with the number of iterations for each neural network validation set is shown in Fig. 4. The total number of iterations for each participating neural network in the comparison is the same.

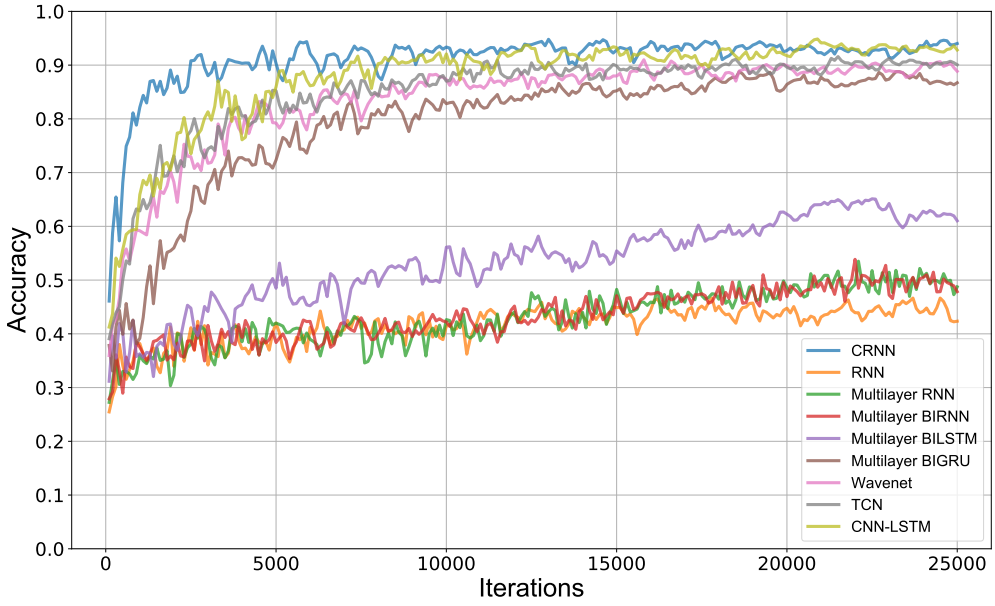

Fig. 4. Curves of validation set classification accuracy with the number of iterations for each neural network in the music genre classification task.

Fig. 4 shows that CNN variants or CNN downsampling-based models generally have a higher fitting ability than other RNN variants under the same model complexity conditions. This is because CNNs can achieve long-term dependency modeling after multiple downsampling, while RNNs are prone to forgetting information and thus find it difficult to establish long-term dependencies. The fitting ability of the CRNN proposed in this study is the highest among all networks (Table 2), which proves that the CRNN can perform a pattern analysis of long-sequence music data well.

Table 2
Performance comparison for music genre classification.

Method	Accuracy	Complexity	Layer
RNN	42.35%	799 MB	1
Multilayer RNN	47.12%	799 MB	6
Multilayer BIRNN	48.01%	799 MB	6
Multilayer BILSTM	61.03%	799 MB	6
Multilayer BIGRU	86.72%	799 MB	6
Wavenet	88.85%	799 MB	6
TCN	90.47%	799 MB	6
CNN-LSTM	94.02%	799 MB	6
CRNN	95.13%	799 MB	6

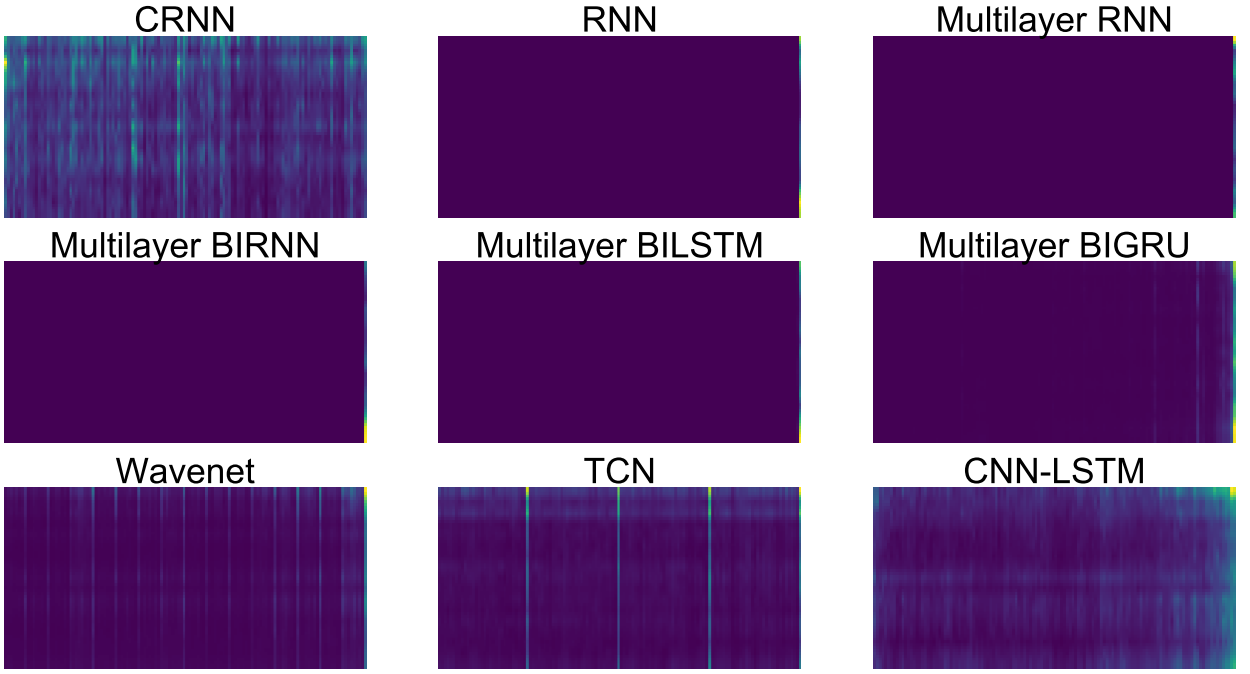

Fig. 5. The gradient amplitude of the output value of each neural network against the input layer in the experiment.

In this experiment, we compute the gradient amplitude of each neural network output value to the input layer (Fig. 5). Fig. 5 shows that the conventional RNN-based improved structures have great gradients only at the end of the sequence, which indicates that they forget most of the information in the sequence and rely only on very little information at the end to make inferences. Although CNN-LSTM has a certain gradient amplitude on the whole, it is obvious that the gradient amplitude at the end of it is much higher than that at the front end, which indicates that there is a large amount of frontend information in the sequence that is forgotten by it. Although WaveNet and TCN also have some high gradient values at the front end of the sequence, there is an obvious periodicity in the location of the high gradient region, the gradient distribution is not uniform, and much information in the low gradient region is forgotten. This indicates that these neural networks overemphasize the information in certain parts of the sequence, and lack the ability to filter out irrelevant information from the entire sequence. Only the gradient amplitude of CRNN is evenly distributed throughout the sequence, which demonstrates the ability of CRNN to maintain long-term memory.

5.2. pMNIST classification

pMNIST is a challenging sequence classification task that flattened all the images in the MNIST dataset into one-dimensional sequences and disrupted them according to a fixed random arrangement. As the original pixel order is disrupted, the proximity pixel dependencies that can be easily modeled are transformed into long-term correlations. Therefore, correct classification can only be achieved if the model remembers the dependencies at different positions in the sequence.

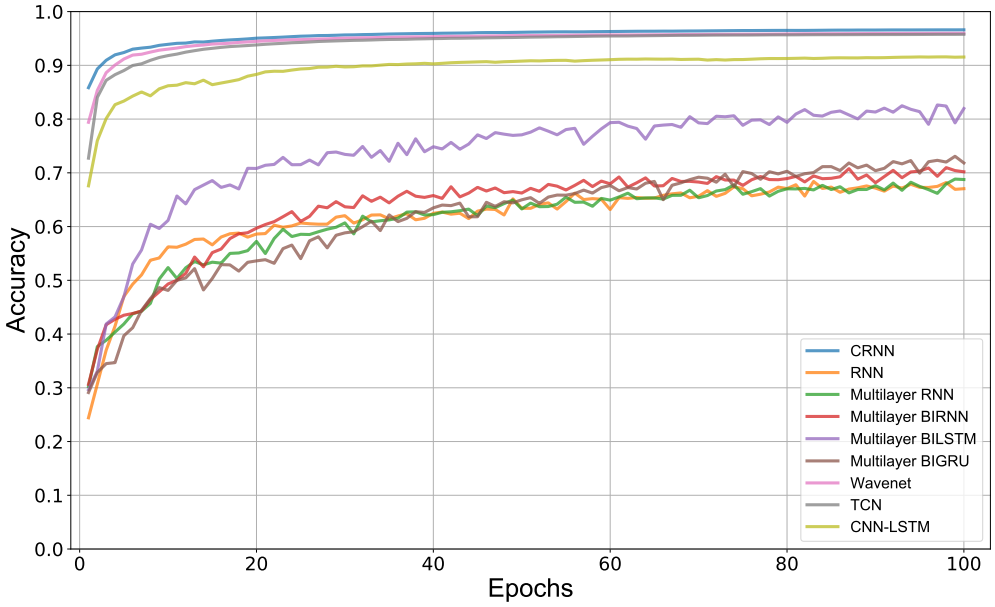

Fig. 6. Curves of test set classification accuracy with the number of epochs for each neural network in the pMNIST classification task.

In our experiments, we use a linear sampling function as the cuneate layer of CRNN with a maximum number of units in the hidden layer of 32. The variation in classification accuracy with the number of iterations for each neural network test set is shown in Fig. 6. In this task, the optimization speed and optimization accuracy of CRNN are higher than all the baselines (Table 3).

Table 3
Performance comparison for pMNIST classification.

Method	Accuracy	Complexity	Layer
RNN	67.05%	268 KB	1
Multilayer RNN	68.75%	268 KB	5
Multilayer BIRNN	70.19%	268 KB	5
Multilayer BILSTM	81.96%	268 KB	5
Multilayer BIGRU	71.84%	268 KB	5
Wavenet	95.96%	268 KB	5
TCN	95.77%	268 KB	5
CNN-LSTM	91.55%	268 KB	5
CRNN	96.69%	268 KB	5

5.3. Bearing remaining useful life prediction

We conducted experiments on industrial signals with longer sequence lengths, and XJTU-SY Bearing Datasets were used to predict the remaining bearing life in this study. We used one of the experimental datasets, operating mode 1, which has five experimental signals. In each experiment, the vibration sensor collected vibration signals of 1.28 s at 1 min intervals (Table 4). Experiments 1-1, 1-2, 1-3, and 1-4 of these experiments were used as training sets for this experiment and validated in Experiments 1-5.

Table 4
XJTU-SY bearing datasets information in operating mode 1.

Signal	Number of Samples	Expected Service Life	Actual Service Life
Bearing 1_1	123		2 h 3 min
Bearing 1_2	161		2 h 41 min
Bearing 1_3	158	5.600~9.677h	2 h 38 min
Bearing 1_4	122		2 h 2 min
Bearing 1_5	52		52 min

We subjected the signals to an STFT transform with a Hann window size of 16 as preprocessing. Five consecutive segments of time-frequency signals with a random intercept in training were used as input, each with a timing length of 16384, based on which the neural network predicted the remaining useful life of the bearings thereafter. We use an attention-based sampling function as the cuneate layer of the CRNN. The mean absolute error (MAE) loss of each neural network in the test set with the number of iterations is shown in Fig. 7. In this experiment, CRNN outperforms the other neural networks (Table 5), which indicates that CRNN has a greater advantage in the recognition task of ultralong mechanical vibration signals.

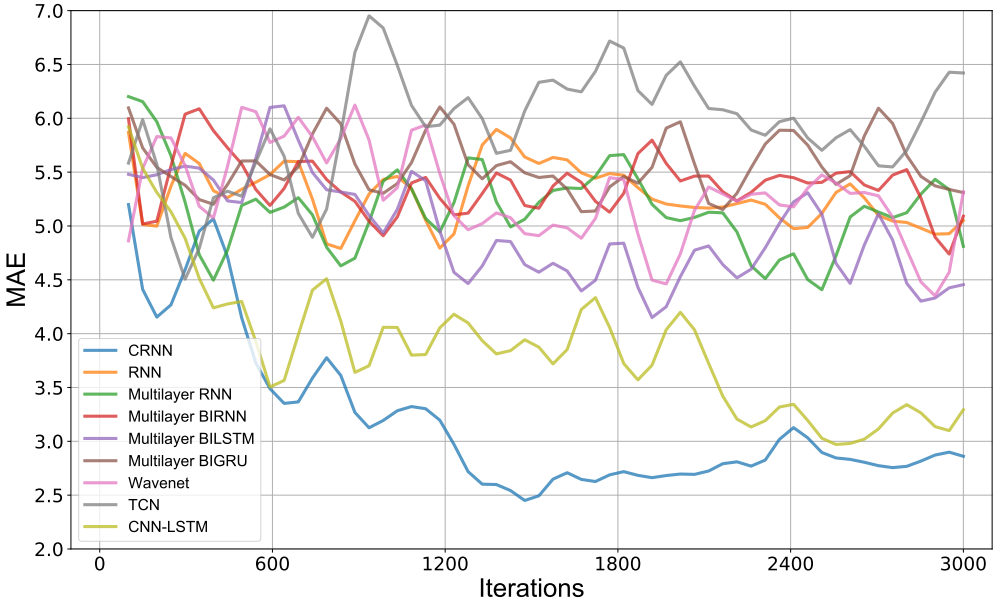

Fig. 7. Curves of MAE loss with the number of iterations for each neural network in the RUL prediction task.

Table 5
Performance comparison for RUL prediction.

Method	MAE	Complexity	Layer
RNN	5.05	70 MB	1
Multilayer RNN	4.81	70 MB	6
Multilayer BIRNN	5.09	70 MB	6
Multilayer BILSTM	4.45	70 MB	6
Multilayer BIGRU	5.31	70 MB	6
Wavenet	5.31	70 MB	6
TCN	6.45	70 MB	6
CNN-LSTM	3.29	70 MB	6
CRNN	2.72	70 MB	6

5.4. Character level WOS-46985 document classification

Document classification is an advanced task in natural language pattern analysis in which the neural network needs to make inferences based on longer texts. WOS-46985 is an interesting document classification dataset that contains abstracts of 46,985 retrievable papers on the Web of Science. The labels are the major and minor categories to which the paper belongs, and there are 134 minor categories in total.

Table 6
Performance comparison for document classification.

Method	Accuracy	Complexity	Layer
RNN	1.22%	430 MB	1
Multilayer RNN	1.34%	430 MB	5
Multilayer BIRNN	1.29%	430 MB	5
Multilayer BILSTM	1.22%	430 MB	5
Multilayer BIGRU	41.75%	430 MB	5
Wavenet	14.80%	430 MB	5
TCN	36.35%	430 MB	5
CNN-LSTM	65.51%	430 MB	5
CRNN	79.53%	430 MB	5

In our experiments, we train the neural network to classify the paper into minor categories. To better compare the performance of various neural networks in ultralong-sequence text analysis, characters are used as the minimum input unit in the experiments. We convert each character in the document into a one-hot code of length 58 and randomly intercept a sequence of characters of length 4096 as the input of the neural network. We use an attention-based sampling function as the cuneate layer of the CRNN. The accuracy variation curve of the validation set in the experiment is shown in Fig. 8, where CRNN is one of the few neural networks that can achieve fitting, and it has the highest fitting speed and accuracy (Table 6). This demonstrates the superior performance of CRNN in the ultralong text analysis task.

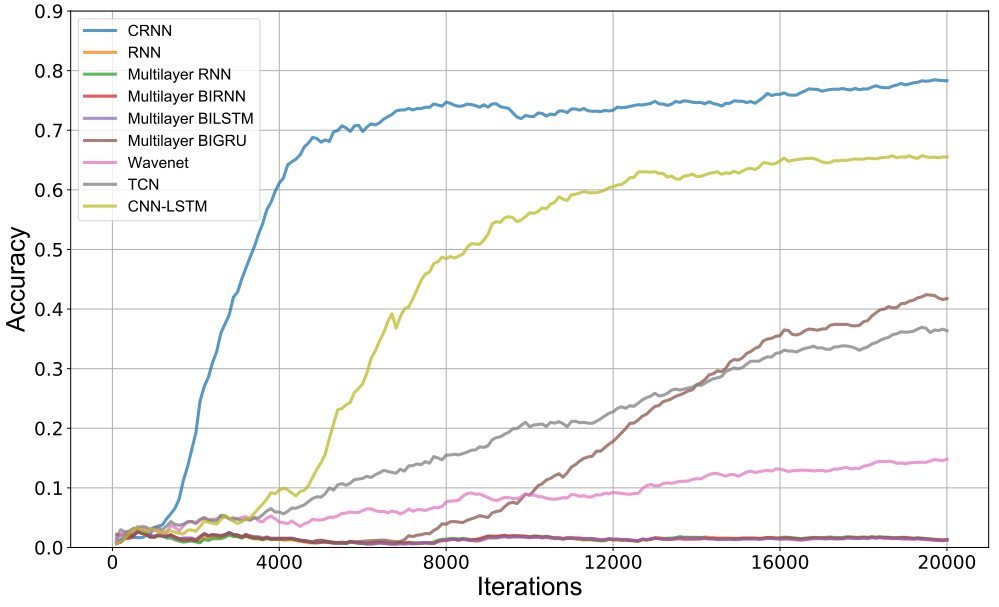

Fig. 8. Curves of test set classification accuracy with the number of iterations for each neural network in the document classification task.

5.5. Character level PTB language modeling

We used Penn Treebank as a dataset for character-level language modeling. In this task, the goal of the neural network is to predict the next character based on the string above. We convert each character in the PTB dataset to a one-hot encoding of length 49 and randomly intercept a sequence of characters of length 1024 as the input to the neural network, and we use bits per character (BPC) [58] as the cost function. Since this is a sequence prediction task, we use the slice sampling function as the cuneate layer of the CRNN. The optimization curves of each network are shown in Fig. 9, where the CRNN has the highest fitting speed and achieves the lowest error (Table 7).

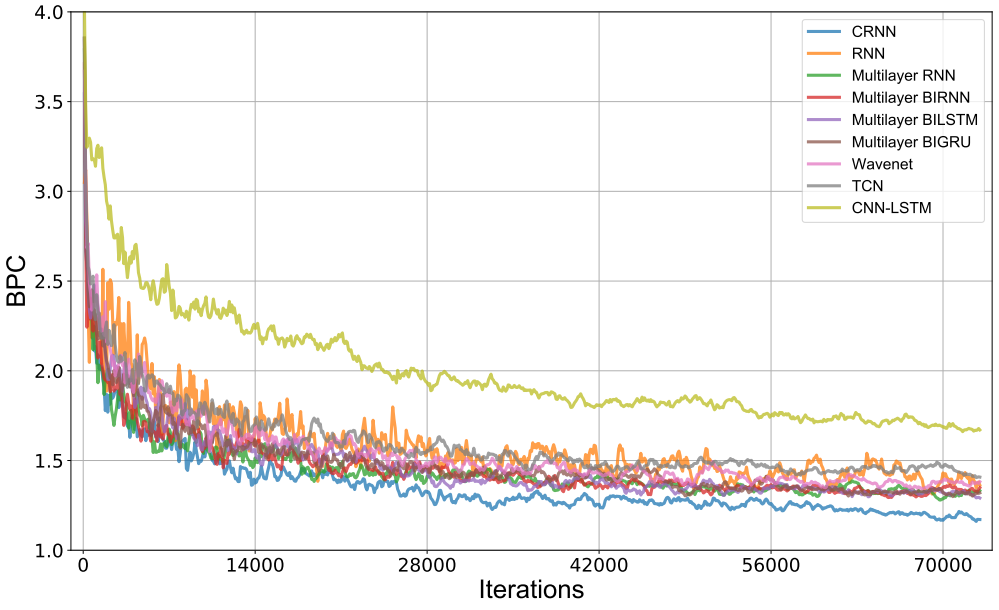

Fig. 9. Curves of BPC loss with the number of iterations for each neural network in the language modeling task.

Table 7
Performance comparison for language modeling.

Method	BPC	Complexity	Layer
RNN	1.412	2.54 GB	1
Multilayer RNN	1.286	2.54 GB	6
Multilayer BIRNN	1.351	2.54 GB	6
Multilayer BILSTM	1.318	2.54 GB	6
Multilayer BIGRU	1.321	2.54 GB	6
Wavenet	1.389	2.54 GB	6
TCN	1.463	2.54 GB	6
CNN-LSTM	1.685	2.54 GB	6
CRNN	1.167	2.54 GB	6

6. Conclusion and future work

We propose CRNN, a novel RNN structure designed to solve the pattern analysis problem for ultralong sequences, which is often very difficult for conventional training processing models. Analysis of the theoretical model shows that both conventional RNN and RNN variants are unable to prevent memory decline and signal interference problems well. As sequence length increases, modeling long-term dependencies becomes increasingly difficult. Inspired by the mechanism of interference signal filtering by cuneate circuit organization in the mammalian brain, we propose the cuneate layer, which can filter interference signals well and shorten the information propagation chain to prevent memory decline. Extensive experimental evaluations of several publicly available datasets confirm that CRNN has better fitting ability and speed for the same model complexity. It can achieve higher accuracy with lower computational costs when computing long-sequence data.

We have new ideas for subsequent research paths. On the one hand, image recognition based on the sequence processing model is one of the current research hotspots [59, 60], and it is promising to develop an image classification recognition framework based on this model. On the other hand, the neural network proposed in this study seems to be theoretically well-suited for building translation or generation frameworks for very long texts, and we will make further attempts in this direction. We will also further optimize this model to improve its applicability and performance.

CRediT authorship contribution statement

Lingkai Hu: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft. **Feng Zhan:** Conceptualization, Methodology, Software, Validation, Formal analysis, Writing – original draft. **Wenkai Huang:** Conceptualization, Methodology, Supervision, Writing – review & editing. **Weiming Gan:** Methodology, Validation, Formal analysis, Investigation, Data curation, Writing – original draft. **Haoxiang Hu:** Formal analysis, Investigation, Data curation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data are available based on a request from the corresponding author.

Acknowledgments

This work was supported by Guangzhou Youth Science and Technology Education Project under Grant KP2024403.

References

[1] Yihao Xue, Rui Yang, Xiaohan Chen, Weibo Liu, Zidong Wang, and Xiaohui Liu. A review on transferability estimation in deep transfer learning. *IEEE Transactions on Artificial Intelligence*, 2024.

[2] Ming Jin, Huan Yee Koh, Qingsong Wen, Daniele Zambon, Cesare Alippi, Geoffrey I Webb, Irwin King, and Shirui Pan. A survey on graph neural networks for time series: Forecasting, classification, imputation, and anomaly detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[3] Shuang Li, Jiangjie Chen, Siyu Yuan, Xinyi Wu, Hao Yang, Shimin Tao, and Yanghua Xiao. Translate meanings, not just words: Idiomkb's role in optimizing idiomatic translation with language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18554–18563, 2024.

[4] Jia Gong, Lin Geng Foo, Yixuan He, Hossein Rahmani, and Jun Liu. Llms are good sign language translators. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18362–18372, 2024.

[5] Chad C Tossell, Nathan L Tenhundfeld, Ali Momen, Katrina Cooley, and Ewart J de Visser. Student perceptions of chatgpt use in a college essay assignment: Implications for learning, grading, and trust in artificial intelligence. *IEEE Transactions on Learning Technologies*, 2024.

[6] Dmitrii Kochkov, Janni Yuval, Ian Langmore, Peter Norgaard, Jamie Smith, Griffin Mooers, Milan Klöwer, James Lottes, Stephan Rasp, Peter Düben, et al. Neural general circulation models for weather and climate. *Nature*, pages 1–7, 2024.

[7] Gregory J Hakim and Sanjit Masanam. Dynamical tests of a deep-learning weather prediction model. *Artificial Intelligence for the Earth Systems*, 2024.

[8] Mostafa M Amin, Rui Mao, Erik Cambria, and Björn W Schuller. A wide evaluation of chatgpt on affective computing tasks. *IEEE Transactions on Affective Computing*, 2024.

[9] Zhige Chen, Rui Yang, Mengjie Huang, Chengxuan Qin, and Zidong Wang. Bdan: Mitigating temporal difference across electrodes in cross-subject motor imagery classification via generative bridging domain. *arXiv preprint arXiv:2404.10494*, 2024.

[10] Chao Li, Ning Bian, Ziping Zhao, Haishuai Wang, and Björn W Schuller. Multi-view domain-adaptive representation learning for eeg-based emotion recognition. *Information Fusion*, 104:102156, 2024.

[11] Xiaohan Chen, Rui Yang, Yihao Xue, Mengjie Huang, Roberto Ferrero, and Zidong Wang. Deep transfer learning for bearing fault diagnosis: A systematic review since 2016. *IEEE Transactions on Instrumentation and Measurement*, 72:1–21, 2023.

[12] Gen Li and Jason J Jung. Deep learning for anomaly detection in multivariate time series: Approaches, applications, and challenges. *Information Fusion*, 91:93–102, 2023.

[13] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

[14] Kyunghyun Cho. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[15] Jürgen Schmidhuber, Sepp Hochreiter, et al. Long short-term memory. *Neural Comput*, 9(8):1735–1780, 1997.

[16] Filippo Maria Bianchi, Enrico Maiorino, Michael C Kampffmeyer, Antonello Rizzi, and Robert Jenssen. An overview and comparative analysis of recurrent neural networks for short term load forecasting. *arXiv preprint arXiv:1705.04378*, 2017.

[17] Adyasha Dash and Kathleen Agres. Ai-based affective music generation systems: A review of methods and challenges. *ACM Computing Surveys*, 56(11):1–34, 2024.

[18] Shih-Lun Wu, Chris Donahue, Shinji Watanabe, and Nicholas J Bryan. Music controlnet: Multiple time-varying controls for music generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:2692–2703, 2024.

[19] Shu Fang, Lijuan Li, Zhenpeng Luo, Zhuangcheng Fang, Dongchao Huang, Feng Liu, Hongliang Wang, and Zhe Xiong. Novel frp interlocking multi-spiral reinforced-seawater sea-sand concrete square columns with longitudinal hybrid frp–steel bars: Monotonic and cyclic axial compressive behaviours. *Composite Structures*, 305:116487, 2023.

[20] Zhuangcheng Fang, Haozhen Fang, Junxing Huang, Haibo Jiang, and Gongfa Chen. Static behavior of grouped stud shear connectors in steel–precast uhpc composite structures containing thin full-depth slabs. *Engineering Structures*, 252:113484, 2022.

[21] Junyu Qi, Rui Zhu, Chenyu Liu, Alexandre Mauricio, and Konstantinos Gryllias. Anomaly detection and multi-step estimation based remaining useful life prediction for rolling element bearings. *Mechanical Systems and Signal Processing*, 206:110910, 2024.

[22] Jiayu Shi, Jingshu Zhong, Yuxuan Zhang, Bin Xiao, Lei Xiao, and Yu Zheng. A dual attention lstm lightweight model based on exponential smoothing for remaining useful life prediction. *Reliability Engineering & System Safety*, 243:109821, 2024.

[23] Yihao Xue, Rui Yang, Xiaohan Chen, Baoye Song, and Zidong Wang. Separable convolutional network-based fault diagnosis for high-speed train: A gossip strategy-based optimization approach. *IEEE Transactions on Industrial Informatics*, 2024.

[24] Zachary C Lipton, John Berkowitz, and Charles Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.

[25] Mehmet Ozgur Turkoglu, Stefano D'Aronco, Jan Dirk Wegner, and Konrad Schindler. Gating revisited: Deep multi-layer rnns that can be trained. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(8):4081–4092, 2021.

[26] Haojin Hu, Mengfan Liao, Chao Zhang, and Yanmei Jing. Text classification based recurrent neural network. In *2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*, pages 652–655. IEEE, 2020.

[27] Ruairidh M Battleday, Joshua C Peterson, and Thomas L Griffiths. From convolutional neural networks to models of higher-level cognition (and back again). *Annals of the New York Academy of Sciences*, 1505(1):55–78, 2021.

[28] Wei Fang, Zhaofei Yu, Zhaokun Zhou, Ding Chen, Yanqi Chen, Zhengyu Ma, Timothée Masquelier, and Yonghong Tian. Parallel spiking neurons with high efficiency and ability to learn long-term dependencies. *Advances in Neural Information Processing Systems*, 36, 2024.

[29] James M Conner, Andrew Bohannon, Masakazu Igarashi, James Taniguchi, Nicholas Baltar, and Eiman Azim. Modulation of tactile feedback for the execution of dexterous movement. *Science*, 374(6565):316–323, 2021.

[30] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[31] Zhongyang Han, Jun Zhao, Henry Leung, King Fai Ma, and Wei Wang. A review of deep learning models for time series prediction. *IEEE Sensors Journal*, 21(6):7833–7848, 2019.

[32] Junyoung Chung. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[33] Víctor Campos, Brendan Jou, Xavier Giró-i Nieto, Jordi Torres, and Shih-Fu Chang. Skip rnn: Learning to skip state updates in recurrent neural networks. *arXiv preprint arXiv:1708.06834*, 2017.

[34] Anupiya Nugaliyadde, Ferdous Sohel, Kok Wai Wong, and Hong Xie. Language modeling through long-term memory network. In *2019 international joint conference on neural networks (IJCNN)*, pages 1–6. IEEE, 2019.

[35] Scott Wisdom, Thomas Powers, John Hershey, Jonathan Le Roux, and Les Atlas. Full-capacity unitary recurrent neural networks. *Advances in neural information processing systems*, 29, 2016.

[36] Quoc V Le, Navdeep Jaitly, and Geoffrey E Hinton. A simple way to initialize recurrent networks of rectified linear units. arxiv 2015. *arXiv preprint arXiv:1504.00941*, 2015.

[37] Eugene Vorontsov, Chiheb Trabelsi, Samuel Kadoury, and Chris Pal. On orthogonality and learning recurrent networks with long term dependencies. In *International Conference on Machine Learning*, pages 3570–3578. PMLR, 2017.

[38] Hossein Abbasimehr, Mostafa Shabani, and Mohsen Yousefi. An optimized model using lstm network for demand forecasting. *Computers & industrial engineering*, 143:106435, 2020.

[39] Jaeyoung Kim, Mostafa El-Khamy, and Jungwon Lee. Residual lstm: Design of a deep recurrent architecture for distant speech recognition. *arXiv preprint arXiv:1701.03360*, 2017.

[40] Ibomoiye Domor Mienye, Theo G Swart, and George Obaido. Recurrent neural networks: A comprehensive review of architectures, variants, and applications. *Information*, 15(9):517, 2024.

[41] Tae-Young Kim and Sung-Bae Cho. Predicting residential energy consumption using cnn-lstm neural networks. *Energy*, 182:72–81, 2019.

[42] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[43] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Trellis networks for sequence modeling. *arXiv preprint arXiv:1810.06682*, 2018.

[44] Md Amirul Islam, Sen Jia, and ND Bruce. How much position information do convolutional neural networks encode? arxiv 2020. *arXiv preprint arXiv:2001.08248*, 2001.

[45] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[46] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[47] Michael Hahn. Theoretical limitations of self-attention in neural sequence models. *Transactions of the Association for Computational Linguistics*, 8:156–171, 2020.

[48] Shengjie Luo, Shanda Li, Shuxin Zheng, Tie-Yan Liu, Liwei Wang, and Di He. Your transformer may not be as powerful as you expect. *Advances in Neural Information Processing Systems*, 35:4301–4315, 2022.

[49] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

[50] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR, 2020.

[51] Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena: A benchmark for efficient transformers. *arXiv preprint arXiv:2011.04006*, 2020.

[52] Tianyang Lin, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. A survey of transformers. *AI open*, 3:111–132, 2022.

[53] George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *IEEE Transactions on speech and audio processing*, 10(5):293–302, 2002.

[54] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[55] Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S Gerber, and Laura E Barnes. Hdltex: Hierarchical deep learning for text classification. In *2017 16th IEEE international conference on machine learning and applications (ICMLA)*, pages 364–371. IEEE, 2017.

[56] Mitch Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993.

[57] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[58] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[59] Jack Lanchantin, Tianlu Wang, Vicente Ordonez, and Yanjun Qi. General multi-label image classification with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16478–16488, 2021.

[60] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.