UDC 334.029.3

DEVELOPMENT OF RECURRENT NEURAL NETWORKS

S.S. WEI

Belarusian State University of Informatics and Radioelectronics, Republic of Belarus

Received March 17, 2023

Abstract. The Recurrent neural network (RNN) has been the main implementation of neural network sequence model, which is the standard processing tool for machine translation, machine question answering, and sequence video analysis, as well as the mainstream modeling tool for problems such as automatic handwriting synthesis, speech processing, and image generation. In the paper, the traditional RNN and the improved Long short-term memory (LSTM) are described in detail.

Keywords: RNN, neural network sequence model, LSTM.

Introduction

Artificial neural networks (ANNs) are algorithms inspired by the neuroscience of the brain, and their basic building blocks are neurons [1]. The nervous system of the brain is the most complex organism in the human body. It consists of nearly 86 billion neurons, each with thousands of synapses connected to other neurons, and together these countless neurons form the brain's complex nervous system. Artificial neurons are abstractly constructed by simulating the functions of biological neurons in the brain, receiving a series of signal inputs and activating them to produce corresponding outputs, and establishing connections between them according to a specific topological network structure.

In the traditional neural network model, it is fully connected from the input layer to the implicit layer to the output layer, and the nodes between each layer are connectionless. However, there is a class of problems that traditional neural networks cannot solve, that is, the training sample input is a continuous sequence, and the length of the sequence varies, such as time-based sequences: a continuous segment of speech, a continuous segment of handwritten text. These sequences are long and of different lengths, so it is difficult to split them into individual samples for training by traditional neural network models. Therefore, a new neural network structure is needed, and thus a recurrent neural network is proposed.

Recurrent neural network is a kind of network model that can accurately model the temporal data. Similar to convolutional neural networks which are good at processing and learning from gridded data, convolutional neural networks are able to process gridded data of variable size, most recurrent neural networks are also able to process sequential data of uncertain length [2-4]. Currently recurrent neural networks have been applied to several fields, especially when there is temporal correlation in the data. The specific structure of recurrent neural networks is that the recurrent neural network will store the memory of previous information in the hidden layer and then input to the currently computed hidden layer unit, the internal nodes of the hidden layer are no longer independent of each other, but have messages to each other.

However, RNN networks still have some shortcomings, such as gradient disappearance and gradient explosion problems. These two situations are usually caused by the backpropagation algorithm. If the gradient disappears, the network layer weights cannot be updated and the training is terminated early, while the gradient explosion causes the network layer parameters to change too much at one-time step and the learning process is unstable. The most intuitive impact of these two problems is that RNNs cannot effectively utilize historical information and are difficult to achieve long-term dependence. The LSTM solves this problem by introducing a gating mechanism to better control the speed of information propagation and thus change the backpropagation structure.

Feedforward neural network

Feedforward neural network (FNN), is a forward-structured neural network. Each neuron is arranged in layers, and each neuron is connected to the neuron of the previous layer only. The output of the previous layer is received and output to the next layer, and there is no feedback between the layers, which is one-way propagation. In this neural network, each neuron can receive signals from the neurons in the previous layer and produce outputs to the next layer. Layer zero is known as the input layer, the last layer is referred to as the output layer, and the other intermediate layers are called hidden layers. The hidden layer can be one layer or multiple layers. Figure 1 shows the structure of FNN. Each node except the input node is a neuron with a nonlinear activation function, which defines a $f(x,\theta)$ mapping that enables the network to achieve the best approximation of the function by learning the value of the parameter θ.

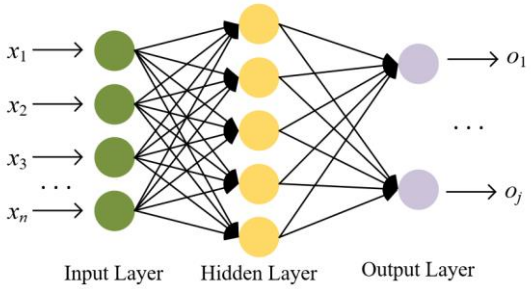

Figure 1. The structure of FNN

A neural network is a nonlinear combination of weighted inputs through an activation function that forms a nonlinear decision boundary, and if the inputs are relevant to the neural network, they are selected for activation, stored, and passed backward. If they are not important, they are suppressed and the inputs do not continue to be passed backwards. The activation functions for neural networks are Tanh, Sigmoid and Relu activation functions, as shown in Table 1.

Table 1. **Three common expressions for activation functions in neural networks**

Activation functions	Function expressions	Derivative expressions
Tanh	$f(x)=\frac{1-e^{-2x}}{1+e^{-2x}}$	$f'(x)=1-f(x)^2$
Sigmoid	$f(x)=\frac{1}{1+e^{-x}}$	$f'(x)=\frac{e^{-x}}{(1+e^{-x})^2}$
Relu	$f(x)=\begin{cases}0, x\le 0\\ x, x>0\end{cases}$	$f'(x)=\begin{cases}0, x\le 0\\ 1, x>0\end{cases}$

Recurrent neural network

RNN are extensions of traditional feedforward neural networks, capable of handling variable-length sequential inputs, which learn the hidden representation of variable-length input sequences by means of internal recurrent hidden variables, the output of the activation function of the hidden variable at each moment depends on the output of the activation function of the recurrent hidden variable at the previous moment [5]. Fig. 2 shows the structure of RNN, x is the input of the model, $t-1$, t, $t+1$ is the time series, s_t is the memory at time *t* after input x_t, which represents the value of the hidden layer; U is the weight matrix from the input layer to the hidden layer; o is the value of the output layer; V is the weight matrix from the hidden layer to the output layer. The weight matrix W is the value of the hidden layer last time as the weight of this input. f and g are both activation functions:

$$s_t = f\left(Ux_t + Ws_{t-1}\right), \tag{16}$$

$$o_t = g\left(Vs_t\right). \tag{17}$$

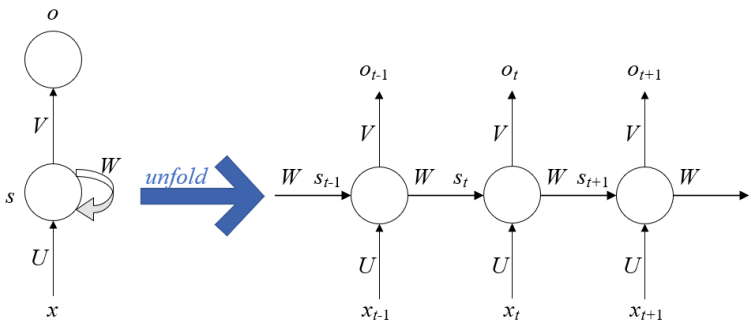

Figure 2. The structure of RNN

The recurrent neural network model is trained by Back Propagation Through Time (BPTT) algorithm, the main use of this algorithm is to process time series, so to be based on time back propagation. The core idea of BPTT algorithm is the same as BP algorithm, is to update the three weight values of W, U and V by gradient descent, so that the error is minimized. The core idea of the BPTT algorithm is the same as that of the BP algorithm. However, RNN can lose gradient after several training sessions, and in some cases, the gradient can be exploded. In response to the difficulty of training recurrent neural networks, various variants of recurrent neural networks have been proposed, such as LSTM, Gated Recurrent Unit (GRU), Bi-LSTM, etc.

Long short-term memory

The main idea of LSTM is to introduce a gate mechanism, so that each LSTM unit can control the degree of historical information retained and remember the current input information, thus capturing potential large-scale sequence dependencies and discarding unimportant features, which are widely used in speech recognition, natural language processing, text compression and handwritten text recognition [6]. A typical LSTM unit is shown in Figure 3.

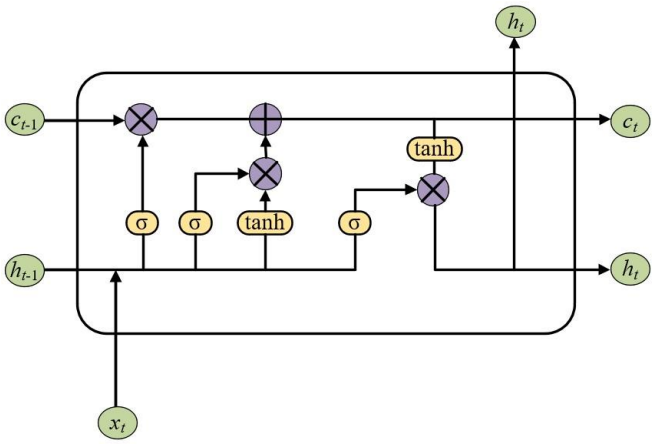

Figure 3. The structure of LSTM

A LSTM unit contains three gates with different functions: an input gate, a forgetting gate and an output gate, which give the LSTM unit the ability to remember information. The input gate determines the proportion of the current state flowing into the current memory cell, the forgetting gate controls the proportion of information in the previous memory cell that is forgotten, and the output gate determines the degree of influence of the current memory cell on the current output.

The information that determines the state of the unit that can be passed. The forgetting gate consists of a sigmoid function that determines the extent to which the unit state can pass based on the output of the previous moment and the input of the current moment. The forgetting gate indicates how much of the state information from the previous time point should be forgotten, which determines how

much of the cell state c_{t-1} from the previous time point is saved to the current cell state c_t. The hidden state h_{t-1} from the previous time point is connected with the input x_t from the current time point to form a new feature quantity, which is then multiplied with the weight parameter W_f, and input to the sigmoid activation function, whose output vector ft is multiplied by the corresponding element of c_{t-1}, $f_t c_{t-1}$ to determine how much of the previous cell state c_{t-1} has been added to the current cell state c_{t-1}. The closer the element of f_t is to 0, the more information in c_{t-1} has been forgotten, and the closer the element of f_t is to 1, the more information in c_{t-1} has been retained. The calculation is shown in equation (3), where W_f and b_f denote the weight parameter and bias term of the sigmoid activation function in the forgetting gate, respectively:

$$f_t = \sigma(W_f \left[h_{t-1}, x_t\right] + b_f). \tag{18}$$

Generate new information that needs to be updated. It consists of two parts, the first step is the input gate to decide which values are used to update, and the second is the tanh function used to generate new candidate values. The input gate indicates how much input information should be remembered at the current point in time, and it determines how much of the input x_t of the cell at the current moment is saved to the cell state c_t. The candidate information c_t at the current moment, determined by the tanh activation function, is determined by multiplying the decision vector it with the corresponding element of the candidate information c_t is $c_t i_t$, to determine how much of c_t is added to the calculation of the unit state c_t. i_t and c_t are shown in equations (4) and (5), respectively. W_i and W_c denote the weight parameters corresponding to sigmoid and tanh in the input gate, respectively, and b_i and b_c denote the corresponding bias terms:

$$i_t = \sigma(W_i \left[h_{t-1}, x_t\right] + b_i), \tag{19}$$

$$c_t = \tanh(W_c \left[h_{t-1}, x_t\right] + b_c). \tag{20}$$

Multiply the previous state value by the coefficient obtained in step 1 to obtain the forgotten part of this unit, and then add the obtained value to the result obtained by multiplying to obtain the new candidate value, which determines the degree of update of each state value:

$$c_t = f_t c_{t-1} + i_t c_t \ . \tag{21}$$

Get the output. An initial output is first obtained by output gating, and then c_t is scaled to between (–1, 1) and then multiply it pairwise with the initial output to obtain the output of the model. The output gate indicates how much of the cell state information should be output at the current point in time, which determines how much of the cell state c_t, at the current moment, is output to the hidden state h_t of the cell. The decision vector o_t of the cell state c_t and the hidden state h_t of the cell, are shown in equations (7) and (8), respectively. W_o and b_o denote the weight parameter and bias term of the sigmoid activation function in the output gate, respectively:

$$o_t = \sigma(W_o \left[h_{t-1}, x_t\right] + b_o), \tag{22}$$

$$h_t = o_t \tanh(c_t). \tag{23}$$

Both the forgetting gate, the input gate and the output gate can be seen to be determined by a value between 0 and 1, which are h_{t-1} and x_t are calculated by the sigmoid function, and although they are in different positions, they measure the necessity of passing the current data backwards: in the forgetting gate, it determines whether c_{t-1} is to be forgotten or not; in the input gate, it determines whether to join or not to join c_t, to update the memory. In the output gate, it determines whether or how important this updated memory is to be input to the next hidden layer.

Conclusion

The development of recurrent neural networks is reviewed and their application background and model characteristics are described in the paper. With the development and fusion of various types of networks, many complex sequence, speech and image problems can be solved, and the variety of recurrent neural networks is quite rich and developing rapidly. With the further deepening of theoretical research and further expansion of application fields, recurrent neural networks will play an increasingly important role.

References

1. Agatonovic Kustrin S., Beresford R. // Journal of pharmaceutical and biomedical analysis. 2000. P. 717–727.
2. Lauriola I., Lavelli A., Aiolli F. // Neurocomputing. 2022. P. 443–456.
3. Jiao M., Wang D., Qiu, J. // Journal of Power Sources. 2020. P. 459.
4. Keren G., Schuller B. // In 2016 International Joint Conference on Neural Networks (IJCNN). 2016. P. 3412–3419.
5. Watson C., Cooper N., Palacio D.N., [et. al] // ACM Transactions on Software Engineering and Methodology (TOSEM). 2022. P. 1–58.
6. Yu Y., Si X., Hu C., [et. al] // Neural computation. 2019. P. 1235–1270.