Neural Networks 145 (2022) 177–188

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

IC neuron: An efficient unit to construct neural networks

Junyi An [a,b], Fengshan Liu [a,b], Furao Shen [a,b,*], Jian Zhao [c,**], Ruotong Li [a,d], Kepan Gao [a,b]

[a] *State Key Laboratory for Novel Software Technology, Nanjing University, China*
[b] *Department of Computer Science and Technology, Nanjing University, China*
[c] *School of Electronic Science and Engineering, Nanjing University, China*
[d] *School of Artificial Intelligence, Nanjing University, China*

A R T I C L E I N F O

Article history:
Received 12 December 2020
Received in revised form 19 May 2021
Accepted 4 October 2021
Available online 23 October 2021

Keywords:
Neural network
Neuron model
Elastic collision
Inter-layer collision neuron

A B S T R A C T

As a popular machine learning method, neural networks can be used to solve many complex tasks. Their strong generalization ability comes from the representation ability of the basic neuron models. The most popular neuron model is the McCulloch–Pitts (MP) neuron, which uses a simple transformation to process the input signal. A common trend is to use the MP neuron to design various neural networks. However, the optimization of the neuron structure is rarely considered. Inspired by the elastic collision model in physics, we propose a new neuron model that can represent more complex distributions. We term it the Inter-layer Collision (IC) neuron which divides the input space into multiple subspaces to represent different linear transformations. Through this operation, the IC neuron enhances the non-linear representation ability and emphasizes useful input features for a given task. We build the IC networks by integrating the IC neurons into the fully-connected, the convolutional, and the recurrent structures. The IC networks outperform the traditional neural networks in a wide range of tasks. Besides, we combine the IC neuron with deep learning models and show the superiority of the IC neuron. Our research proves that the IC neuron can be an effective basic unit to build network structures and make the network performance better.

1. Introduction

Artificial neural network (ANN) is an important branch of machine learning. The McCulloch–Pitts (MP) neuron (McCulloch & Pitts, 1990) is the basic unit of ANN and the most commonly used neuron structure. As shown in Fig. 1(a), the MP neuron can be formulated as $y = \mathrm{f}(\sum_{i=1}^{n} w_i x_i + b)$, where a linear transformation and a non-linear activation function are applied to the input successively. This structure enables the network to represent the non-linear distributions of the input signal (Zhang, Bengio, Hardt, Recht, & Vinyals, 2017). Currently, most researchers focus on changing the connections between the MP neurons to develop new network structures, of which fully-connected (FC) neural network, convolutional neural network (CNN) (Krizhevsky, Sutskever, & Hinton, 2017) and recurrent neural network (RNN) (Graves, Mohamed, & Hinton, 2013) are the most popular ones. Based on these network structures, deep learning (DL) leads the trend of stacking a large number of artificial neurons to solve complex tasks (Krizhevsky, Sutskever, & Hinton, 2012a; Simonyan & Zisserman, 2015). However, the developing of new neuron models has not drawn much attention.

Biological neuron is the major reference for neuron remodeling. The MP neuron proposed by simply imitating the behavior of biological neurons is far from reaching the full potential of a biological neuron (Ostojic & Brunel, 2011). Therefore, some work tries to introduce more complex biological mechanisms to artificial neurons. Recently, the neurotransmitter transmission mechanism has drawn more attention. We know that a neuron transmits information to a neighboring neuron through releasing neurotransmitters. The neurotransmitters will be blocked or strengthened before being caught by the neighboring neurons (Debanne, Campanac, Bialowas, Carlier, & Alcaraz, 2011). The dendritic neuron model (DNM) (Todo, Tang, Todo, Ji, & Yamashita, 2019) used the sigmoid function to directly process the received inputs, treating the inputs as neurotransmitters which are selectively received. Several theoretical research (Cazé, Jarvis, Foust, & Schultz, 2017; Gao et al., 2019) have inferred that by taking into account the synaptic nonlinearities in a dendritic tree, the DNM could be an effective calculative unit. Another popular study of imitating biological mechanisms is spiking neural networks (SNNs) (Eliasmith et al., 2012). The SNNs aimed to model the sparse asynchronous voltage pulses which control the transmission of neurotransmitters. The unique temporal

* Corresponding author at: Department of Computer Science and Technology, Nanjing University, China.
** Corresponding author at: School of Electronic Science and Engineering, Nanjing University, China.
E-mail addresses: dz1833001@smail.nju.edu.cn (J. An), fengshanliuf@gmail.com (F. Liu), frshen@nju.edu.cn (F. Shen), jianzhao@nju.edu.cn (J. Zhao), 181220030@smail.nju.edu.cn (R. Li), gaokp@smail.nju.edu.cn (K. Gao).

https://doi.org/10.1016/j.neunet.2021.10.005
0893-6080/© 2021 Elsevier Ltd. All rights reserved.

dynamics and energy efficiency of SNNs make them develop separately from ANN, and the advances of ANNs have little impact on SNNs (Woźniak, Pantazi, Bohnstingl, & Eleftheriou, 2020).

The above two types of studies usually used some morphological changes to determine the types of transmitter signals that other neurons receive. However, the effects of neurotransmitters may be enhanced by the human body or specific drugs, except for being blocked. We find that the selective enhancement of neurotransmitter has similarities with the elastic collision model in physics which indicates that the information carried by the object can be enhanced through collision. Besides, although there is a spiking neural unit (SNU) (Woźniak et al., 2020) proposed to integrate neural dynamics of SNNs into ANN units, the SNU and DNM cannot be stacked flexibly like what MP neurons do in DL. This is mainly because they have complex non-linear structures. Flexible and simple non-linear structure not only makes neuron representations more continuous, but also facilitates the expansion of neural networks. The single multiplicative neuron (SMN) model (Yadav, Kalra, & John, 2007) and pi-sigma unit (Shin & Ghosh, 1991) proposed the single neuron models with polynomial representations through multiplication of parallel elements. However, they cannot be easily extended as the MP neurons (Egrioglu, Yolcu, Aladag, & Bas, 2015; Lyutikova, 2018), because the multiplication of elements will make the model unrobust in the high dimension spaces. We find that the mathematical form of the elastic collision model keeps the flexibility of non-linear representation and does not introduce the multiplication of parallel elements. Based on the advantages of the elastic collision model, we believe that the elastic collision model has the potential to guide the design of neuron models, and develop a new artificial neuron model, which we name as the Inter-layer Collision (IC) neuron.

The IC neuron imitates the process of elastic collision, transforming the input signal like the way of changing the state of objects during a collision. In addition to the physical characteristics, we give our mathematical analysis to show how the IC neuron works. The structure of the IC neuron is depicted in Fig. 1(b), where w' denotes the adjustment weight which is introduced to adjust a new input branch and σ denotes a non-linear transformation with Rectified Linear Unit (ReLU) (Nair & Hinton, 2010). When an input signal passes through the $+$ operation, the input space is divided into multiple subspaces which can represent different linear transformations. Moreover, the IC neuron retains the advantages of the linear representation ability of the MP neuron. In particular, it retains the independence of connection weights. Both the IC neuron and MP neuron can map the input with any dimension to a one-dimensional (1-D) output. We prove that in a finite input space, the IC neuron can perfectly represent all the distributions that the MP neuron can represent with more fine-grained features. Based on these observations, we believe that the IC neuron can replace MP neuron and learn more complex distributions of the input signals.

Since the IC neuron has the same form of input and output as the MP neuron, we can easily integrate it into existing network structures, including the classical FC, the convolutional, and the recurrent structures. As shown in Fig. 1(c), the frameworks of the three networks with the IC neurons are similar to classical networks. The IC neuron can improve the generalization ability of the networks by optimizing the non-linear representation ability of the basic computing unit without changing the network connections. Besides, unlike some models that use non-linear kernel functions as basic units to improve their non-linear representation ability (Mairal, Koniusz, Harchaoui, & Schmid, 2014; Wang, Yang, Xie, & Yuan, 2019), the networks with the IC neurons do not map the inputs to a high-dimensional space, making it only requires a small amount of calculation and parameters, which is close to classical networks. We design experiments for IC networks with the three connection structures. Our experiments show that, in a variety of classification tasks, the IC networks not only exceed the classical networks in accuracy with the same hyperparameters but also show the advantage of accelerating the training process. Furthermore, the IC networks can achieve better accuracy than the classical networks with a larger scale, which shows that the IC neuron is an effective basic building unit for neural networks. Besides, we show the IC neuron can be easily integrated into some existing DL models. This method reduces the difficulty of designing the network by bypassing the complicated hyperparameter design. The DL experiments show the superiority and universality of IC neurons on complex tasks.

Our contribution can be summarized as follows:

- Inspired by the physical collision model, we propose the IC neuron as a basic building block for neural networks. We prove that the IC neuron can represent more complex distributions compared to the commonly used MP neuron.
- The IC neuron we propose is a basic computational unit that can be integrated into the majority of existing network structures. We propose strategies to combine the IC neuron with popular network structures, such as the FC, the convolutional and the recurrent structures. Our research shows that using the IC neuron and the proposed strategies can build more efficient networks.

The rest of the paper is organized as follows. We introduce the details of the IC neuron and the networks with IC neurons in Section 2. We construct a series of experiments to investigate the effectiveness of the IC neuron in Section 3. Finally, we summarize this study in Section 4.

2. The Inter-layer collision model

The IC neuron is inspired by the elastic collision model. In this section, we first describe a physical collision scene that follows the laws of the conservation of momentum and energy. We then introduce the IC neuron and its physical characteristics. Section 2.3 gives a mathematical analysis of why the IC neuron has a stronger representation ability than the MP neuron. Finally, we build the networks with the IC neurons and analyze the time and space complexity of them.

2.1. The physical elastic collision scene

Physical models often reveal the update or transfer process of physical quantities, e.g., the elastic collision model can clearly explain the transfer of momentum between objects. Consider an ideal physical environment with only collision forces. Two objects move in a 1-D space where we use the left $(-)$ and the right $(+)$ to indicate the direction. The mass of them are m_1 and m_2, respectively. Initially, m_1 lies to the left of m_2 and they both have a velocity of zero. When m_1 is given a positive speed v_1 toward m_2, according to the laws of energy conservation and momentum conservation, the velocity of m_1 and m_2 after collision are:

$$v_1' = \left(\frac{2m_1}{m_1 + m_2} - 1\right) v_1,\ v_2' = \frac{2m_1}{m_1 + m_2} v_1. \tag{1}$$

According to the above analysis, the result of this collision process is that the object m_2 will always move to the right $(+)$ while the direction of the object m_1 depends on the quantitative relationship between m_1 and m_2. To observe this result, we can place an observation platform on the far right of the above physical scene to measure the speed value. In another way, we can treat this physical model as an information transmitting system where we only care about the overall information flowing out of

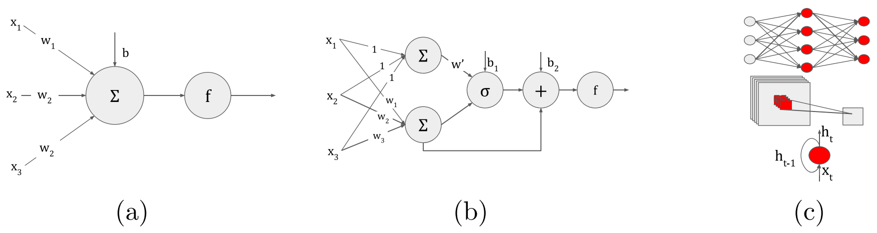


Fig. 1. (a) The MP neuron. x_i denotes the input signal, w_i denotes the connection weight and b denotes the bias; (b) The IC neuron. σ denotes a non-linear operation, w' denotes an adjustment weight, $+$ denotes addition and f denotes activation operation; (c) The FC, the convolutional and the recurrent structures with the IC neurons. The red shapes represent the computational structure of the IC neurons. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

the system given the input information. In this case, the input information is v_1 and the output information is $v'' = v_1'' + v_2''$. Here v_1'' and v_2'' are given by:

$$v_1'' = \sigma\left((w-1)v_1\right), v_2'' = wv_1, \tag{2}$$

where w denotes the coefficient $\frac{2m_1}{m_1+m_2}$, $\sigma(\cdot)$ denotes the ReLU function which is used to get the right component (+) of v_1', i.e., capture objects that continue to move to the right. We notice that w acts as a parameter controlling how much information could be transmitted. In the way, if we make w learnable like what we do in machine learning and add this structure into the neural network framework, the system can be tuned to transmit useful information to the subsequent layers. Based on this assumption, we build a learnable model. To standardize the notation, we replace the (v_1, v'') with (x, y) to denote the input and output, respectively. The mathematical form is given by:

$$y = wx + \sigma\left((w-1)x\right). \tag{3}$$

Inspired by this propagation model, we build the basic IC computational unit in the next section.

2.2. The basic IC computational unit

The process of velocity quantity transmission system formulated in Eq. (3) represents a single input and single output model. When the single unit receives multiple inputs, we define the output by adding the results of each input, which can be viewed as the accumulation of multiple collision results on the same object. We combine this process with the activation operation. Considering a n-D input and a bias term, the mathematical form is given by:

$$\begin{aligned} y &= \mathrm{f}\left(\sum_{i=1}^{n} w_i x_i + \sigma\left(\sum_{i=1}^{n}(w_i-1)x_i + b_1\right) + b_2\right) \\ &= \mathrm{f}\left(\sum_{i=1}^{n} w_i x_i + \sigma\left(\sum_{i=1}^{n} w_i x_i - x_{sum} + b_1\right) + b_2\right), \end{aligned} \tag{4}$$

where $\mathrm{f}(\cdot)$ is used to denote an activation function, such as sigmoid or ReLU function. b_1 and b_2 are two independent bias used to adjust the center of the model distribution. $x_{sum} = \sum_{i=1}^{n} x_i$ represents the summation of all the input features. Eq. (4) defines the basic version of the IC neuron. In the physical scene constructed in 2.1, we can determine whether the object m_1 moves to the right observation platform by controlling the quantitative relationship between m_1 and m_2. This can be regarded as a sifting process that observers always hope to measure components that carry key information instead of being disturbed by irrelevant ones. Corresponding to the IC neuron, the input signal is divided into $\sum_{i=1}^{n} w_{ij}x_i + b_2$ and $\sigma\left(\sum_{i=1}^{n}(w_i-1)x_i + b_1\right)$. The term $\sigma\left(\sum_{i=1}^{n}(w_i-1)x_i + b_1\right)$ can be considered as the physical characteristics of the object m_1, which enhances the key information and suppresses the useless ones. In addition to fitting features, IC neuron can play a role in controlling the flow of information, capturing fine-grained features through a multilayer framework, which is beneficial for a given task. In order to further improve the non-linear representation ability of the basic IC neuron, we relax the constant 1 in terms of $\sigma\left(\sum_{i=1}^{n}(w_i-1)x_i + b_1\right)$ to a learnable value w'. The formulation is given by:

$$\begin{aligned} y &= \mathrm{f}\left(\sum_{i=1}^{n} w_i x_i + \sigma\left(\sum_{i=1}^{n}(w_i-w')x_i + b_1\right) + b_2\right) \\ &= \mathrm{f}\left(\sum_{i=1}^{n} w_i x_i + \sigma\left(\sum_{i=1}^{n} w_i x_i - w'x_{sum} + b_1\right) + b_2\right), \end{aligned} \tag{5}$$

where w' is named as the adjustment weight. It can be regarded as the intrinsic weight of one neuron, which is different from the weight w_i connecting two neurons. We term Eq. (5) the standard IC neuron, which not only retains the physical significance of the elastic collision model but also adds more flexibility in controlling the information transmission. We prove that the introduction of w' can alleviate the restriction on non-linear representation.

2.3. The mathematical analysis of the IC neuron

Multiple linear transformation. We first consider the representation ability of the basic IC neuron. Here we omit the bias term to simplify the notation. Different from the traditional neuron $y = f(\sum_{i=1}^{n} w_i x_i)$, Eq. (4) uses a non-linear representation instead of a linear one inside the activation function f. We use the term $H = \sum_{i=1}^{n}(w_i - w')x_i$ to represent a hyperplane ($H = 0$) in an N-dimensional Euclidean space, which divides Eq. (4) as follows:

$$y = \begin{cases} \mathrm{f}\left(2\sum_{i=1}^{n} w_i x_i - \sum_{i=1}^{n} x_i\right) & \text{if } H \geq 0 \\ \mathrm{f}\left(\sum_{i=1}^{n} w_i x_i\right) & \text{if } H < 0. \end{cases} \tag{6}$$

Intuitively, this IC neuron has a stronger representation ability than the MP neuron, since it can produce two different linear representations instead of one before the activation operation $\mathrm{f}(\cdot)$. To visually distinguish the difference between the two kinds of neurons, we use 2-D data as input and ReLU as activation function to show how the two kinds of neurons generate non-linear boundaries. Fig. 2(a, b) shows that the 2-D Euclidean space is divided into multiple subspaces by the two neurons, where each subspace represents a linear transformation or a zero mapping. We observe that a single IC neuron can add an extra subspace to represent a different linear transformation. Furthermore, we map the XOR problem which is a typical linear inseparable problem onto a 2-D plane (Fig. 2(c)) to explain the difference of the two kinds of neurons. To distinguish between $(1, 0), (0, 1)$ and $(1, 1), (0, 0)$, the four points need to be divided into at least three linear subspaces. For the ReLU MP neuron, the non-linear boundary is a straight line. It is clear that at least two neurons are required to solve the XOR problem. However, the non-linear

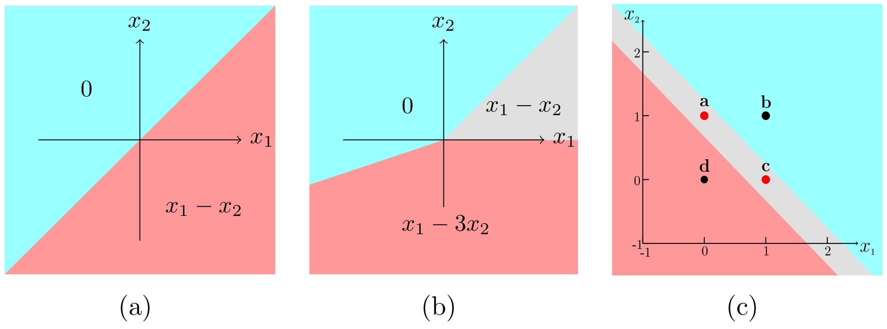


Fig. 2. The value of $f(x_1, x_2)$ given x_1, x_2. (a): $f(x_1, x_2) = \sigma(x_1 - x_2)$. (b): $f(x_1, x_2) = \sigma(x_1 - x_2 + \sigma(-2x_2))$. (c): $f(x_1, x_2) = \sigma(w_1x_1 + w_2x_2 + b_1 + \sigma((w_1 - 1)x_1 + (w_2 - 1)x_2 + b_2))$. Here $w_1 = w_2 = 0.2805$. $b_1 = -0.3506$ and $b_2 = 0.6463$ are used to shift the boundary across the whole space.

boundary of the IC neuron is a broken line, providing a single neuron possibility to solve the XOR problem. Fig. 2(c) gives a solution by using a single IC neuron, dividing the plane into three subspaces where $(0, 0), (1, 1)$ are represented by zero and $(1, 0), (0, 1)$ are in two spaces with similar representation.

Adjustment weight. Although the basic version of the IC neuron uses the hyperplane $H = 0$ to increase the number of non-linear patterns, the position of the hyperplane is limited by weight w_i, making it not flexible to generate different subspaces. Fig. 3(a) shows the formation of a hyperplane in a 3-D space where $\mathbf{W} = (w_1, w_2, w_3)$ denotes the connection weights, $\mathbf{I} = (1, 1, 1)$ and $w' = 1$ is a constant in the basic version of the IC neuron. The main functionality of $\mathbf{W}$ is to learn the distribution of different input features instead of the hyperplane $H = 0$. When the weight $\mathbf{W}$ is fixed, the angle between the hyperplane $H = 0$ and the coordinate system is also fixed. In this case, the subspaces separated by the hyperplane are usually not optimal and the learning of weights easily converges to a local minimum. To add more representation flexibility to the hyperplane $H = 0$, the standard IC neuron uses an adjustment weight w'. Then there are two independent parameters controlling the position of the hyperplane: w' is used to change the angle between the hyperplane and the coordinate system, and the bias b_1 is used to shift the hyperplane in the whole space. Fig. 3(b) shows how the angle of the hyperplane changes after adding the learnable adjustment weight. For higher dimensions, we give a theoretical explanation of the adjustable range of w':

Theorem 1. *By adjusting w', the hyperplane $\sum_{i=1}^{n}(w_i - w')x_i = 0$ can be rotated π rad around the cross product of vector $\mathbf{W}$ and $\mathbf{I}$ when two vectors are linearity independent. Here $\mathbf{W} = (w_1, w_2, \dots, w_n)^T$ and $\mathbf{I} = (\underbrace{1, 1, \dots, 1}_{n})^T$.*

Proof. The normal vector of $\sum_{i=1}^{n}(w_i - w')x_i = 0$ is given by $\mathbf{H} = (w_1 - w', w_2 - w', \dots, w_n - w')^T$. Consider the angle between $\mathbf{H}$ and $\mathbf{I}$:

$$\cos(\theta) = \frac{\mathbf{H}^T \cdot \mathbf{I}}{|\mathbf{H}||\mathbf{I}|} = \frac{\mathbf{W}^T \cdot \mathbf{I} - Nw'}{\sqrt{N|\mathbf{W}|^2 - 2Nw'\mathbf{W}^T \cdot \mathbf{I} + N^2w'^2}}. \tag{7}$$

When $\mathbf{W}^T \cdot \mathbf{I} \geq Nw'$:

$$\cos(\theta) = \sqrt{1 + \frac{(\mathbf{W}^T \cdot \mathbf{I})^2 - N|\mathbf{W}|^2}{N|\mathbf{W}|^2 - 2Nw'\mathbf{W}^T \cdot \mathbf{I} + N^2w'^2}}. \tag{8}$$

Eq. (8) is a continuous subtractive function, and $\cos(\theta) \in [0, 1)$ when $w' \in (-\infty, \frac{\mathbf{W}^T \cdot \mathbf{I}}{n}]$. Similarly, $\cos(\theta) \in (-1, 0]$ when $w' \in [\frac{\mathbf{W}^T \cdot \mathbf{I}}{n}, \infty)$. Therefore, we get $\theta \in (0, \pi)$ when $w' \in (-\infty, \infty)$. It is clear that the direction of the rotation axis is the same as the cross product of $\mathbf{W}$ and $\mathbf{I}$. □

Theorem 1 implies that $\sum_{i=1}^{n}(w_i - w')x_i = 0$ can almost represent the whole hyperplanes parallel to the cross product of $\mathbf{W}$ and $\mathbf{I}$, providing flexible strategies for dividing spaces. In summary, by adjusting the relationship between $\mathbf{W}$ and w', the IC neuron can not only retain the representation ability of the MP neuron but also flexibly generate linear representation spaces for some complex distribution.

2.4. Building the IC networks

We further build neural networks by using the standard IC neuron. Due to the characteristic that the IC neuron can transform the input with any size to 1-D output, we integrate the IC neuron into most popular neural network structures, such as the fully-connected, the convolutional, and the recurrent structures.

Fully-connected structure. First, we consider building a fully-connected layer with the IC neuron. The input dimension is n and the number of neurons is m, and the output of the jth neuron can be expressed as:

$$y_j = \mathrm{f}\left(\sum_{i=1}^{n} w_{ij}x_i + \sigma\left(\sum_{i=1}^{n} w_{ij}x_i - w'_j x_{sum}\right)\right), j \in [1, m]. \tag{9}$$

To simplify the notation, bias terms are omitted here and below. Eq. (9) can be redefined by matrices and vectors as:

$$\mathbf{y} = \mathrm{f}\left(\mathbf{W}_{n\times m}\mathbf{x} + \sigma\left(\mathbf{W}_{n\times m}\mathbf{x} - \mathbf{w}'_{m\times 1}\mathbf{I}_{1\times n}\mathbf{x}\right)\right), \tag{10}$$

where $\mathbf{I}$ is defined as the transpose of a n-D all-one vector for calculating the x_{sum}. $\mathbf{W}$ is the connection weights and $\mathbf{w}'$ is a vector representing adjustment weights. We use Eq. (10) to build the fully-connected IC network, similar to the architecture of multilayer perceptron (MLP). By hierarchically connecting the IC neurons, the IC fully-connected (IC-FC) network can represent more complex non-linear distributions.

Recurrent structure. The classical RNNs use deterministic transitions from previous to current hidden states (Zaremba, Sutskever, & Vinyals, 2014). Given an input sequence $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_T)$, an RNN computes the sequence of h-D hidden vectors $\mathbf{h} = (\mathbf{h}_1, \dots, \mathbf{h}_T)$ and sequence of output vectors $\mathbf{y} = (\mathbf{y}_1, \dots, \mathbf{y}_T)$ by iterating the equations:

$$\begin{aligned} \mathbf{h}_t &= \mathrm{f}(\mathbf{W}_{n\times h}\mathbf{x}_t + \mathbf{W}_{h\times h}\mathbf{h}_{t-1}), \\ \mathbf{y}_t &= \mathbf{W}_{h\times m}\mathbf{h}_t, \end{aligned} \tag{11}$$

where $\mathbf{W}$ denotes the weight matrices and $\mathrm{f}(\cdot)$ denotes the activation function and subscript $t \in [1, T]$ denotes the tth iteration step. We use the IC structure to transform the current input $\mathbf{x}_t$ instead of using $\mathbf{W}_{n\times h}$ and the hidden vector $\mathbf{h}_t$ is reformulated as:

$$\mathbf{h}_t = \mathrm{f}\left(\mathbf{W}_{n\times h}\mathbf{x}_t + \sigma(\mathbf{W}_{n\times h}\mathbf{x}_t - \mathbf{w}'_{h\times 1}\mathbf{I}_{1\times n}\mathbf{x}_t) + \mathbf{W}_{h\times h}\mathbf{h}_{t-1}\right). \tag{12}$$

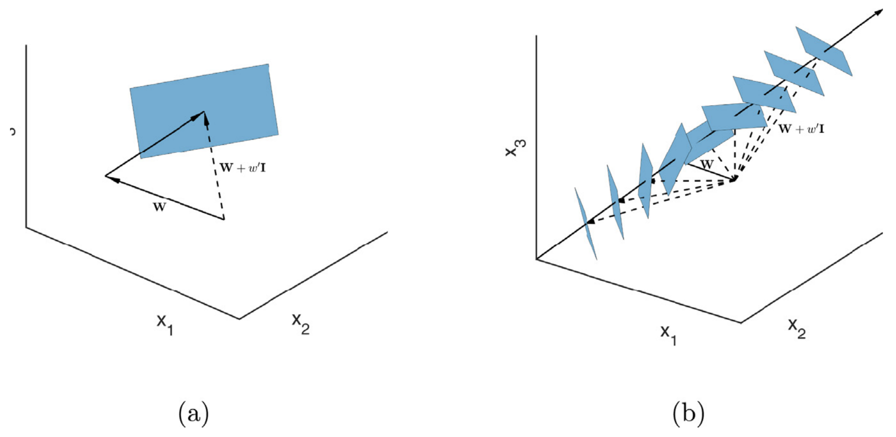


Fig. 3. (a): The hyperplane $H = 0$ dividing the 3-D input space; (b): The rotation process of the hyperplane according to the adjustment weight w'.

The rest of the recurrent structure remains unchanged. Since Eq. (12) retains the recurrent structure, it can learn the previous information in the sequence. Besides, the hidden state $\mathbf{h}_t$ emphasizing the tth input $\mathbf{x}_t$ will make the future context easier to learn. We use Eq. (12) to build the IC recurrent neural network (IC-RNN), which can process sequential data.

Convolutional structure. The convolutional structure uses kernels and sliding window to capture the feature in local area (Krizhevsky et al., 2017; Valueva, Nagornov, Lyakhov, Valuev, & Chervyakov, 2020), mapping an input feature map $\mathbf{X} \in \mathbb{R}^{H' \times W' \times C'}$ to an output $\mathbf{U} \in \mathbb{R}^{H \times W \times C}$. To simplify the notation, the activation operator is omitted. We can write the output feature maps as $\mathbf{U} = [\mathbf{U_1}, \mathbf{U_2}, \ldots, \mathbf{U_C}]$, where

$$\mathbf{U}_i = \mathbf{W}_i * \mathbf{X}. \tag{13}$$

Here $\mathbf{W}_i \in \mathbb{R}^{k \times k \times C'}$ is one filter kernel and $*$ denotes the convolutional operator. We retain the characteristics of sliding window which takes advantages of the shared weights, then using Eq. (5) to replace the calculation in the kernel in each window. the new kernel is represented by $[\mathbf{W_i}, w_i']$:

$$\begin{aligned}\mathbf{U_i} &= [\mathbf{W}_i, w_i'] * \mathbf{X} \\ &= \mathbf{W}_i * \mathbf{X} + \sigma(\mathbf{W}_i * \mathbf{X} - w_i' \times (\mathbf{I} * \mathbf{X})),\end{aligned} \tag{14}$$

where $\mathbf{I}$ used to sum the local area is an all-one tensor with the same size as $\mathbf{W}_i$. Since the IC convolutional layer defined by Eq. (14) enjoys the stronger non-linear representation ability of the IC neuron, it can capture more abstract local features without changing the form of the input and output feature maps. We stack this structure to build the IC convolutional neural network (IC-CNN).

2.5. Parameters and complexity analysis

Parameters. We compare the standard IC neuron with the MP neuron in three network structures. For the IC fully-connected and the recurrent structures, the addition of parameters depends completely on the adjustment weight w' and the bias b_1. If there are m hidden neurons with n-D input, the IC networks will add parameters from $(n + 1) \times m$ to $(n + 3) \times m$. The ratio of addition is $\frac{2}{n+1}$, which can be regarded as adding 2 dimensions to the input. The result of the convolutional structure is similar in that the addition depends on the number of kernels. The real-world tasks usually need hundreds of neurons or kernels, making the effects of additional parameters of the IC neuron negligible.

Computational complexity. The computational complexity of neural networks mainly comes from matrix operations. In the IC fully-connected and the recurrent structures, the $\mathbf{Wx}$ term including $n \times m$ multiplications is used twice but only calculated once. $\mathbf{w'Ix} = \mathbf{w'}(\mathbf{Ix})$ needs $n + m$ multiplications. Compared to the hidden layer with MP neurons. The additional complexity is approximately $\frac{1}{n} + \frac{1}{m}$ of it. For the IC convolutional structure, $\mathbf{W}_i * \mathbf{X}$ is also calculated once. Note that we can first generate a tensor to represent $w_i' \times \mathbf{I}$, and $w_i' \times (\mathbf{I} * \mathbf{X}) = (w_i' \times \mathbf{I}) * \mathbf{X}$ can be regarded as adding a kernel for the input features. So the additional complexity is approximately $\frac{1}{C}$ of classical convolutional structure. We do not change the operations outside the neuron model, including batch normalization (BN), pooling, etc. When n, m or C are relatively large, the complexity of the IC networks is similar to networks with MP neurons.

3. Experiments

In this section we evaluate the performance of the IC networks by integrating the IC neurons into three network structures (IC-FC network, IC-RNN and IC-CNN). We also evaluate the basic version of the IC neuron (Eq. (4)) to investigate the effectiveness of the σ operation and the adjustment weight w'. The corresponding models are IC-FC-B network, IC-RNN-B and IC-CNN-B.

3.1. Configurations and datasets

Our goal is to compare the IC networks with the traditional ones. For fair comparison, we evaluate the traditional networks with the same network hyperparameters as the IC networks. Specifically, we use the ReLU as the activation function f, cross entropy as the loss function and Adadelta as the optimizer. The evaluation metrics include the top-1 accuracy, FLOPs and the amount of parameter. The accuracy of each round is recorded to trace the training process. During the training process, we observe that the test accuracy fluctuates sharply. So we make a simple smoothing of the test curve in order to better visualize the trend of the test accuracy. We repeat each experiment three times and record the best accuracy in order to reduce the influence of random initialization.

We reference the evaluation methods of Zhou and Feng (2017) to evaluate the IC networks. We use a variety of classification tasks. For each task, we individually set the number of layers and hidden neurons. The details are shown below.

Music classification task uses the GTZAN dataset that contains 1000 music clips divided into 10 classes evenly. Each clip is stored in a 30 s long file. We split the dataset into 710 clips for training and 290 clips for testing. In addition, we use MFCC, chroma and some other features to represent each music clip, which transform the original sound wave into a 253-D feature vector. We build the IC-FC network with an input-256-128-output structure.

Table 1
The accuracy of three FC models on several classification tasks.

Model	MNIST	IMDB	GTZAN	sEMG	LETTER	YEAST	ADULT
FC	98.02	87.59	54.14	40.00	93.25	59.19	84.99
IC-FC	**98.42**	**88.28**	**56.21**	**41.67**	**94.00**	**61.88**	**85.20**
IC-FC-B	98.37	88.13	55.17	41.11	93.50	60.54	85.14

Hand movement recognition task uses the UCI sEMG dataset that contains 1800 records divided into six kinds of hand movements, i.e., spherical, tip, palmar, lateral, cylindrical and hook. The sEMG dataset is a time-series dataset, where EMG sensors capture 500 features per second and each record associates with 3000 features. We build the IC-FC network with an input-1024-512-output structure to process the records. We use the IC-RNN with an input-80-output structure to recurrently process each EMG features of the records.

Sentiment classification task uses the IMDB dataset that contains 25 000 movie reviews for training and 25 000 for testing. The reviews are preprocessed by extracting the *tf-idf* features and are cropped into 5000-dimensional features of equal length. We build the IC-FC network with an input-1024-512-256-output structure. Besides, the IMDB dataset contains text information used to evaluate the IC-RNN and RNN. We build a dictionary based on all reviews to obtain the time-series data. Then the word embedding (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013) operation transforms each word into a 128-D vector. We use the IC-RNN with an input-128-output structure to recurrently process each word features of the reviews.

Image classification task uses the MNIST and CIFAR10 dataset. The MNIST contains 60 K training images of 28×28 size and 10 K test images. We evaluate the IC-FC network with an input-256-128-output structure. The IC-CNN for MNIST consists of three 3×3 IC convolutional layer with 32, 64, 64 output channels and 1, 2, 2 strides, respectively. After the IC convolutional layers capture the features, we use a features-256-output traditional FC structure as the classifier. We further use the CIFAR10 dataset to evaluate this IC-CNN. The CIFAR10 consists of 60 K $32 \times 32 \times 3$ RGB images in 10 classes divided into 50 K training images and 10 K testing images.

Besides, we also evaluate some simple IC networks on the UCI YEAST, ADULT and LETTER datasets, which have only 8/14/16 dimensional inputs. YEAST has 1038/446 training/test examples. LETTER has 16 000/4000 training/test examples. ADULT has 32 561/16 281 training/test examples. We build the IC-FC networks with input-32-16-output, input-32-16-output, input-64-32-output structures for YEAST, ADULT and LETTER datasets, respectively. Note that the variance of ADULT data is relatively big, so we put a BN operation at the beginning of the networks.

3.2. Comparative experiment

3.2.1. IC-FC networks

Table 1 records the performance of the IC-FC networks and the corresponding FC networks on the above tasks except the CIFAR10 dataset. We have three major observations. First, the IC networks achieve highly competitive performance in all the tasks. Remarkably, the IC networks have an obvious improvement when the accuracy of the original MP networks is low, such as GTZAN (2.07%), sEMG (1.67%) and YEAST (2.69%). More importantly, we observe from Table 2 that the IC-FC networks add little computational cost. The most obvious additions of FLOPs/Params are the results on the YEAST (8.16%/9.09%) and ADULT (8.65%/9.62%) dataset. According to our analysis in Section 2.5, the influence of additional calculation and parameters will be trivial as the number of hidden-layer neurons increases. This is distinctly proven by

Table 2
The FLOPs/Params of three FC models on several classification tasks. The FLOPs (KMac) denote the amount of calculation for feedforward propagation. The params (K) reflect the amount of learnable parameters.

Model	MNIST	IMDB	GTZAN	sEMG
FC	235.14/235.15	5777.66/5777.67	99.20/99.21	3600.90/3600.92
IC-FC	236.56/235.91	5785.99/5781.25	100.09/99.98	3606.46/3603.97
IC-FC-B	236.18/235.53	5784.20/5779.46	99.71/99.59	3604.92/3602.44
Model	LETTER	YEAST	ADULT	–
FC	4.00/4.03	0.98/0.99	1.04/1.04	–
IC-FC	4.18/4.22	1.06/1.08	1.13/1.14	–
IC-FC-B	4.08/4.12	1.02/1.03	1.09/1.09	–

Table 3
The accuracy/FLOPs/Params of the IC-RNNs and RNNs on time-series tasks.

Model	sEMG	IMDB
RNN	29.07/240.48/240.57	80.64/662.46/662.56
IC-RNN	**31.48**/243.64/240.73	**82.86**/672.91/662.69
IC-RNN-B	30.93/243.56/240.65	79.12/672.85/662.62

the experiments on IMDB and sEMG datasets, which bring negligible additional cost compared to the FC networks (0.14%/0.06% and 0.15%/0.08%). In summary, the IC-FC networks show better representation ability with little additional computational burden. Besides, we observe that the standard version of the IC neuron has better performance compared to the basic ones in most tasks. This result shows that the adjustment weight w' is effective through joint training with other learnable parameters.

The training processes of the IC-FC and the FC networks are depicted in Fig. 4. It is clear that the IC networks converge in a shorter time, indicating that the IC neurons are more conducive to fitting features. Specially, although both the IC networks and FC networks use the same initialization and optimizer, the IC networks fit the distributions of the training datasets better in the first few rounds. When the training accuracy is overly high, the improvements of the test curves are not obvious in the GTZAN and sEMG datasets. We argue that this is because of overfitting. Although the IC neuron helps to find the fine-grained features, the type of these features is similar to the ones obtained from the MP neurons. Besides, two versions of the IC neuron show similar convergence speed, showing that the strong representation ability comes from the σ non-linear operation of the IC neuron. Moreover, the networks with the standard IC neurons achieve higher accuracy, which shows that the adjustment weight w' can improve the upper bound of the representation ability.

3.2.2. IC-RNN and IC-CNN

We build the IC-RNNs to process time-series data, including the sENG and IMDB datasets. Since the dropout does not work well with RNNs and large RNNs tend to overfit (Zaremba et al., 2014), we build the IC-RNNs and the traditional RNNs with small scale. The results, as shown in Table 3 and Fig. 5, show that the IC-RNNs achieve much higher accuracy, demonstrating that the IC neurons can work well with the recurrent structure. We believe that the IC neuron can improve complex recurrent models, including RNNs with multiple hidden layers and LSTM (Sak, Senior, & Beaufays, 2014), etc. We also build the IC-CNNs to process image data, such as the MNIST and CIFAR10 datasets. In order to evaluate the IC neuron in the convolutional structure, the IC-CNNs use the traditional fully-connected layers as the classifiers instead of the IC-FC. The results of the IC-CNNs recorded in Table 4 achieve a higher accuracy, implying that the IC neurons can learn more fine-grained local features. The training curves shown in Fig. 6 show a faster convergence speed of the IC-CNNs. These comparison results show that the IC neuron works well on basic

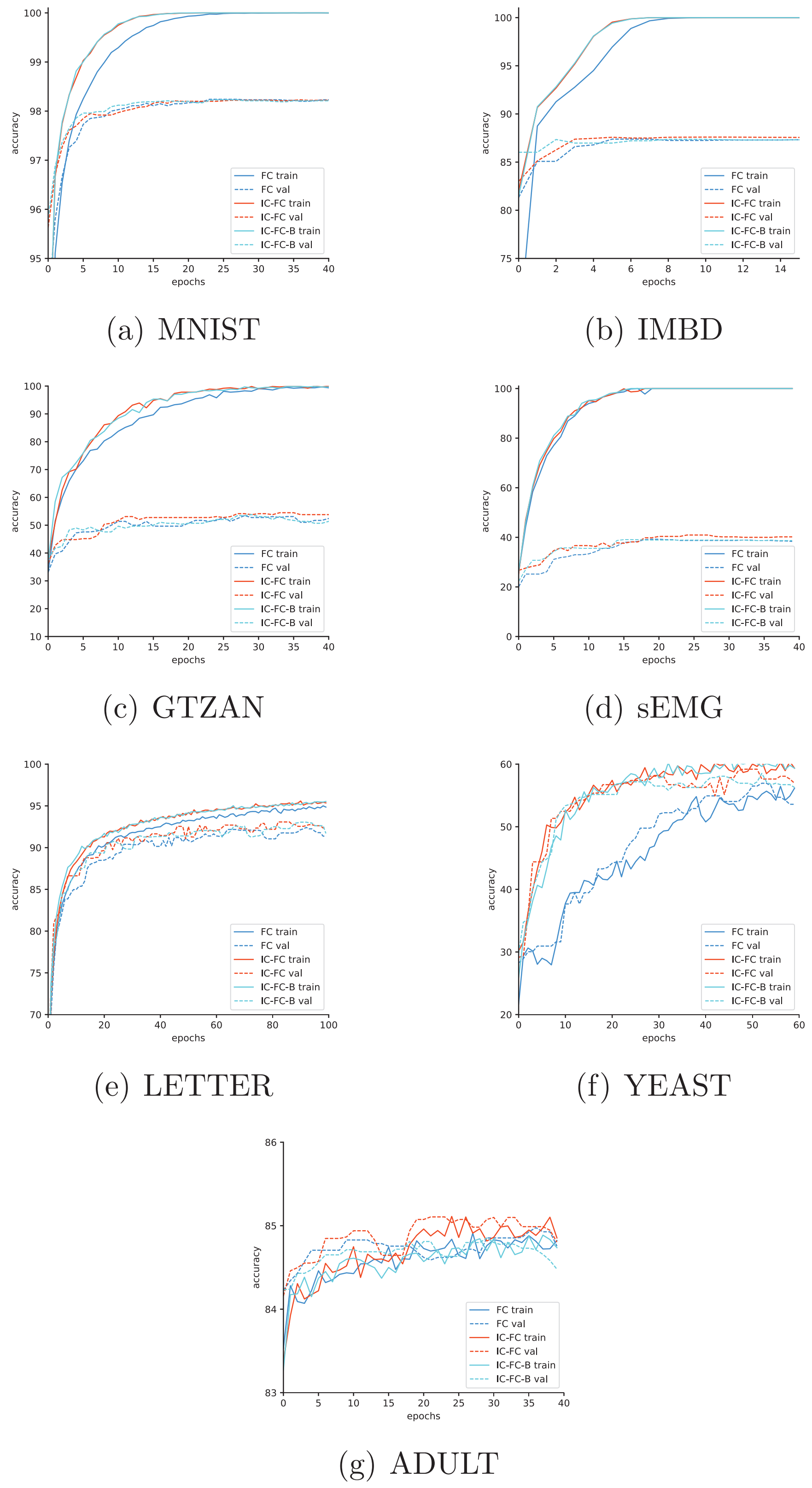
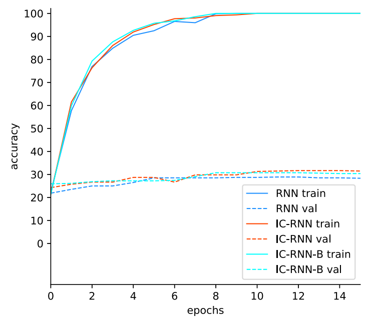


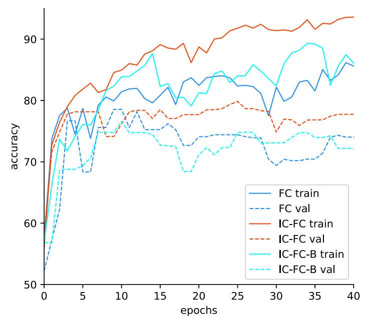
Fig. 4. The training curves of the IC-FC networks and the FC networks on several tasks.

convolutional structures. Similar to the conclusion of the IC-RNNs, we believe that the IC neuron is a useful unit to build bigger convolutional networks (He, Zhang, Ren, & Sun, 2016; Simonyan & Zisserman, 2015). Examples of deep learning with the IC structure shown in Section 3.6 also verify our statement.

In summary, our experiments show that the combination of the IC neuron and the three computational structures is successful. The IC networks can capture fine-grained features, enabling themselves to achieve a higher accuracy and a faster convergence speed. Moreover, they only increase a small computational cost

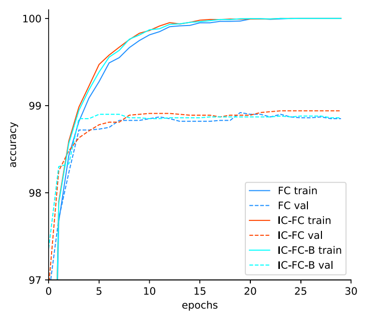


(a) sEMG

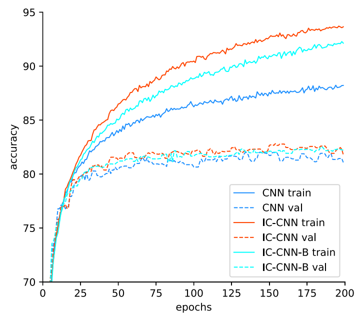


(b) IMDB

Fig. 5. The training curves of the IC-RNNs and the RNNs on time-series tasks.



(a) MNIST



(b) CIFAR10

Fig. 6. The training curves of the IC-CNNs and the CNNs on image tasks.

Table 4
The accuracy/FLOPs/Params of the IC-CNNs and CNNs on image tasks.

Model	MNIST	CIFAR10
CNN	98.84/6491.20/861.39	81.25/9067.26/1107.72
IC-CNN	**98.97**/6623.70/861.71	**83.02**/9258.75/1108.05
IC-CNN-B	98.95/6582.93/861.55	82.82/9205.50/1107.88

Table 5
The accuracy of the IC-FC networks, deeper and wider FC networks on several classification tasks.

Model	MNIST	IMDB	GTZAN	sEMG	LETTER	YEAST	ADULT
Deeper FC	**98.55**	87.98	55.52	40.37	93.56	60.54	85.14
Wider FC	98.53	87.74	55.17	40.56	93.70	60.31	84.99
IC-FC	98.42	**88.28**	**56.21**	**41.67**	**94.00**	**61.88**	**85.20**

Table 6
The FLOPs/Params of the IC-FC networks, deeper and wider FC networks on several classification tasks.

Model	MNIST	IMDB	GTZAN	sEMG
Deeper FC	798.46/798.47	6811.89/6811.91	526.59/526.60	4650.5/4650.50
Wider FC	535.81/535.82	9504.01/9504.04	263.94/263.95	6008.5/6008.51
IC-FC	236.56/235.91	5785.99/5781.25	100.09/99.98	3606.46/3603.97
Model	LETTER	YEAST	ADULT	–
Deeper FC	8.19/8.22	2.03/2.04	2.1/2.1	–
Wider FC	8.08/8.11	6.45/6.46	6.65/6.65	–
IC-FC	4.18/4.22	1.06/1.08	1.13/1.14	–

compared to the classical networks. We think that using the IC neuron is more efficient than expanding the scale of networks. To prove our conjecture, we compare the IC networks with bigger classical networks. Besides, we further investigate the factors affecting the training process. The details are shown in the subsequent chapters.

3.3. Model capacity and performance

We construct experiments to prove the conjecture that using the IC neuron is better than simply increasing the number of parameters. To consider both high-dimensional and low-dimensional data, we evaluate the IC-FC networks on all the datasets listed in Table 1. The bigger fully-connected networks are: input-512-1256-output and input-256-256-128-output for MNIST; input-1500-1000-500-output and input-1024-1024-512-256-output for IMDB; input-512-256-output and input-256-256-128-output for GTZAN; input-1500-1000-output and input-1024-1024-512-output for sEMG; input-100-50-output and input-32-32-16-output for LETTER; input-100-50-output and input-32-32-16-output for YEAST; input-100-50-output and input-32-32-16-output for ADULT. We name them deeper or wider FC networks. The results are recorded in Tables 5 and 6. We observe that expanding the scale of networks is inefficient. Although the deeper and wider FC networks use much more parameters and FLOPs, their results are still worse than the IC networks. Remarkably, the wider FC network for YEAST cost six times computational source as the IC-FC network (6.45 kMacs/6.46 kB vs. 1.06 kMacs/1.08 kB) while its accuracy is still lower than the IC-FC network (60.76% vs. 61.04%). Some work states that large-scale networks training is slow (Ioffe & Szegedy, 2015) as what we observe from the experiments that expanding the scale has no benefits for the convergence speed.

In order to further verify that the generalization abilities of the IC neurons are not from the increased parameters, we evaluate the IC-CNN and ICRNN with a similar method. We use the wider CNN with 32, 64, 128 outputs channels and deeper CNN with 32, 64, 64, 64 outputs channels for MNIST and CIFAR10; wider RNN with input-256-output structure for IMDB and sEMG. We do not

Table 7
The accuracy/FLOPs/Params of the IC-RNNs, IC-CNNs and their corresponding deeper and wider networks on several classification tasks.

Model	sEMG	IMDB
Wider RNN	29.26/770.60/770.31	80.86/1317.67/1317.85
IC-RNN	**31.48**/243.64/240.73	**82.86**/672.91/662.69
Model	MNIST	CIFAR10
Deeper CNN	**99.00**/8300.67/898.31	81.81/11 430.66/1144.65
Wider CNN	98.92/9103.49/1701.13	81.31/12 479.23/2193.23
IC-CNN	98.97/6623.70/861.71	**83.02**/9258.75/1108.05

Table 8
The accuracy of the IC-FC networks with different activation functions.

Function	Model	MNIST	IMDB	GTZAN	sEMG	LETTER	YEAST	ADULT
Tanh	FC	98.23	87.84	53.79	31.72	93.20	60.54	83.96
	IC-FC	**98.31**	**88.52**	**54.48**	**33.52**	**93.45**	**61.88**	**84.19**
ELU	FC	**98.55**	88.18	54.48	34.44	92.93	60.31	83.88
	IC-FC	98.52	**88.24**	**55.52**	**36.48**	**93.65**	**61.66**	**84.24**

construct deeper RNNs because they will lead to overfitting. As shown in Table 7, the IC networks exceed these deeper and wider models with less computational cost. Although the deeper CNN for MNIST has a small improvement compared to the IC-CNN, the increase on FLOPs/Params is not negligible. In summary, the IC networks show superior performance.

3.4. Activation function

We construct experiments to investigate the influence of different activation operations. We have mentioned that the f of the IC neuron can be popular activation functions. However, the neuron model may be hard to train when the non-linear function is complex. We investigate the effects of the commonly used functions, including Tanh (Nwankpa, Ijomah, Gachagan, & Marshall, 2018) and ELU (Clevert, Unterthiner, & Hochreiter, 2016). They can represent smooth boundaries and negative values. From Table 6, we observe that the IC networks have an improvement with the above activation functions. These experiments show that the IC neuron is flexible and universal to replace the MP neuron in many commonly used networks (see Table 8).

3.5. Combination with the MP-based neuron models

We have shown that the core of the IC neuron is replacing the linear transformation with a non-linear one, which lets the IC neuron possess a stronger generalization ability than the MP neuron. Therefore, we conjecture that other neuron models with linear components may benefit from the idea of the IC neuron. The IC neuron can be easily combined with the models with the MP neurons so that we try to integrate it into some MP-based neuron models which use linear weights to transform the inputs, including the Pi-Sigma neural network (PSNN) and the single multiplicative neuron (SMN) model. Note the DNM and SNU are not MP-based neuron models (Gao et al., 2019; Woźniak et al., 2020). Their linear components are intertwined into complex non-linear structures.

PSNN. The PSNN (Shin & Ghosh, 1991) is a higher order neural network that uses the product of the hidden neuron outputs. Its output is given by:

$$y = \mathrm{f}(\prod_{j=1}^{H} h_j),\ h_j = \sum_{i=1}^{N} w_{ij}x_i + b_j, \tag{15}$$

where x_i is the input, w_{ij} and b_j are weight and bias of the PSNN. h_j denotes the neuron in hidden layer and f denotes a suitable activation function. The term H is a hyperparameter denoting the number of hidden neurons. Although there are some studies that tried to optimize the order and combination of the product of hidden neurons or the training methods (Li, 2003; Nayak, Naik, & Behera, 2014), the improvement of linear parts is less considered. Inspired by Eq. (3), we integrate the σ operation and adjustment weight w' into the representation of h_j:

$$h_j = \sum_{i=1}^{n} w_{ij}x_i + \sigma(\sum_{i=1}^{n} w_{ij}x_i - w'x_{sum} + b_{1j}) + b_{2j}, \tag{16}$$

where b_{1j} and b_{2j} are two independent biases. Since the linear part of PSNN is similar to the FC networks, Eq. (16) can take advantages of multiple linear transformations mentioned in Section 2.3.

SMN. The SMN first proposed in Hussain, Liatsis, Tawfik, Nagar, and Al-Jumeily (2008) is based on a polynomial structure. Its output is given by:

$$\begin{aligned} y &= \mathrm{f}(u) \\ u &= \prod_{i=1}^{n}(w_i x_i + b_i), \end{aligned} \tag{17}$$

where x_i is the input of the SMN, w_i and b_i are the weight and bias of the SMN. f denotes an activation function. We apply the σ operation and adjustment weight w' on the term $(w_i x_i + b_i)$, since it represents the linear transformation on the inputs. The formula of u is changed to:

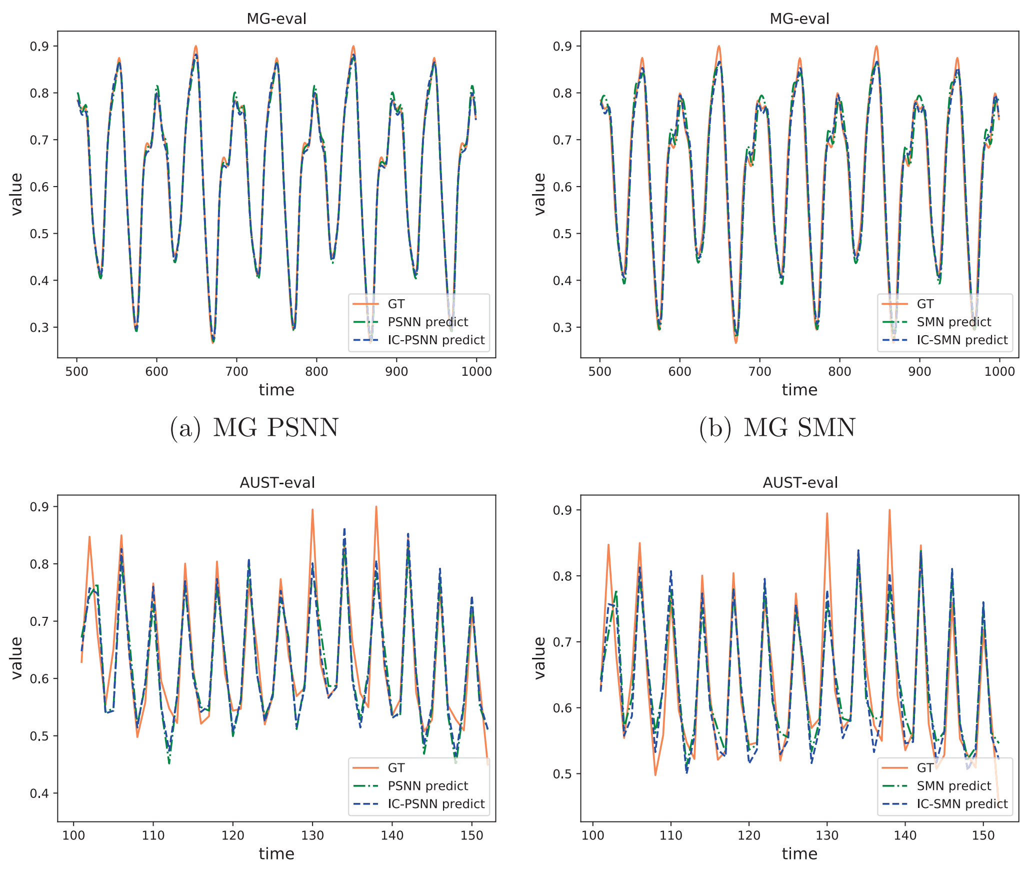
$$u = \prod_{i=1}^{n}(w_i x_i + \sigma((w_i - w')x_i + b_{1i}) + b_{2i}). \tag{18}$$

This structure increases the number of patterns in polynomial representation. Through Eqs. (16) and (19), the PSNN and SMN can take advantage of the multiple linear transformations of the IC neuron. When the extra patterns of the inputs are useless, we can return to traditional SMN and PSNN by adjusting the w' and b_1 terms, similar to the analysis in the Section 2.3.

We use the time series forecasting widely used in Attia, Sallam, and Fahmy (2012), Hussain et al. (2008) and Yadav et al. (2007) to evaluate PSNN and SMN with the IC structure. we use the Mackey–Glass (MG) (Mackey & Glass, 1977) and the Australia beer consumption (AUST) (Janacek, 2001) datasets. The MG time series used for white blood cell production in leukemia patients with non-linear oscillations is widely used for testing the performance of neural network models. The MG delay-difference function is given by:

$$y(t+1) = (1-b)y(t) + a\frac{y(t-\tau)}{1+y^{10}(t-\tau)}, \tag{19}$$

where $a = 0.2$, $b = 0.1$ and $\tau = 17$. The objective of the modeling is to predict the value of the time series based on four previous values (Using $y(t)$, $y(t-6)$, $y(t-12)$, $y(t-18)$ to predict $y(t+1)$). We generate 1000 samples and divide into training set and test set evenly. To facilitate the training of weights, the datasets are normalized between 0.1 and 0.9. The number of hidden neurons of PSNN is set to 5. We use the Adadelta optimizer to train the models in 500 epochs. Fig. 7(a, b) show the prediction results of the IC models and their basic models. We observe that the PSNN, SMN and their corresponding IC models can approximately represent the MG delay-difference function. Remarkably, the IC models tend to focus more on detailed relationship between current data and historical data, especially on the peaks of MG data. In Table 9, we observe that the PSNN and SMN with the IC neurons achieve smaller mean square error (MSE) and mean absolute error (MAE) compared to the basic PSNN and SMN.



(a) MG PSNN (b) MG SMN

(c) AUST PSNN (d) AUST SMN

Fig. 7. Prediction results for the MG time series dataset and the AUST dataset.

Table 9
Comparison of performance for MG and AUST time series dataset between the PSNN, SMN and their corresponding IC models.

Dataset	Model	Testing MSE	Testing MAE	Model	Testing MSE	Testing MAE
MG	PSNN	0.0001	0.0096	SMN	0.0008	0.0234
	IC-PSNN	0.0001	**0.0073**	IC-SMN	**0.0006**	**0.0205**
AUST	PSNN	0.0024	0.0399	SMN	0.0023	0.0363
	IC-PSNN	**0.0019**	**0.0339**	IC-SMN	**0.0017**	**0.0334**

The AUST dataset, a real-world time series dataset that reported the beer consumption in Australia from 1956 to 1994, has 154 samples. We use the first 100 samples for training and the last 54 samples for verification. The experimental configuration is the same as the MG experiment, except that we use $y(t)$, $y(t-1)$, $\dots$, $y(t-6)$ to predict $y(t+1)$. Fig. 7(c, d) shows the prediction results on AUST dataset. We observe that the IC models show better prediction results on the troughs of AUST, like the region from 130 to 140. The MSE and MAE recorded in Table 9 show that the representative ability of PSNN and SMN can be strengthened by introducing the idea of the IC neuron.

3.6. Applications on deep learning

CIFAR. We have shown that the superiority of the IC neuron in some shallow neural networks. To further investigate the applicability of the IC neuron, we integrate the IC structure into some widely-used deep learning (DL) models. Our first set of experiments is using the CIFAR10/CIFAR100 datasets to verify the universality of the IC neuron in some modern CNN architectures, including VGGNets (Simonyan & Zisserman, 2015) (VGG-16 version), MoblieNet (Howard et al., 2017), DenseNet (Huang, Liu, van der Maaten, & Weinberger, 2017) (DenseNet-121 version), ResNets (He et al., 2016) (ResNet-18 and ResNet-50 versions, they use two kinds of blocks), ResNeXt (Xie, Girshick, Dollár, Tu, & He, 2017) (ResNeXt-50 2 $\times$ 64d version) and SENets (Hu, Shen, & Sun, 2018) (SE-ResNet-18 and SE-ResNet-50 versions). Note that the CIFAR100 divides the images with the same size as the CIFAR10 into 100 classes. The optimizer uses the stochastic gradient descent (SGD) (LeCun et al., 1989) method with a weight decay of 10^{-4} and a momentum of 0.9. Each model is trained in 200 epochs with a batch size of 128. The learning rate is initialized to 0.1, which will be reduced 10 times at the 60th and the 120th epoch.

Similar to the IC-CNN experiments, we only replace the traditional convolutional layer with the IC convolutional structure in these models. Especially, the adjacent convolution layers in MoblieNet (a depthwise convolution layer and a pointwise convolutional layer) are treated as a single convolutional layer to be combined with the IC structure since there is a close relationship between adjacent layers. The comparison of top-1 accuracy is shown in Table 10, through which we observe that the IC networks surpass their basic networks by 0.24%/0.77% on average. Especially, the IC-MoblieNet obviously improves the performance of the MoblieNet whose accuracy is not as saturated as other models (from 90.08%/65.98% to 90.90%/67.32%). The IC neuron improves the performance even though these deep architectures already have strong generalization abilities. These experiments show that the effectiveness and universality of the IC structure on deep systems.

Table 10
Results on CIFAR10 with various IC models. We use our environment settings to reproduce the baseline results.

Model	CIFAR10	CIFAR100	Model	CIFAR10	CIFAR100
VGG-16	93.64	72.93	ResNet-18	95.02	75.61
IC-VGG-16	**93.90**	**73.35**	IC-ResNet-18	**95.20**	**75.95**
DenseNet-121	95.41	77.01	ResNet-50	94.94	77.74
IC-DenseNet-121	**95.52**	**79.15**	IC-ResNet-50	**95.12**	**78.59**
MobileNet	90.08	65.98	SE-ResNet-18	94.92	76.44
IC-MobileNet	**90.90**	**67.32**	IC-SE-ResNet-18	**95.06**	**76.60**
ResNeXt	95.38	77.77	SE-ResNet-50	94.90	78.58
IC-ResNeXt	**95.46**	**78.09**	IC-SE-ResNet-50	**95.02**	**79.18**

Table 11
Performance results of the IC networks with 1×1 IC layers on ImageNet validation set.

Model	Top-1 acc./Top-5 acc.	GFLOPs/Params
ResNet-50	76.15/92.87	4.12/25.56M
IC-ResNet-50	**76.73/93.31**	4.14/25.60M
ResNet-101	77.37/93.54	7.85/44.55M
IC-ResNet-101	**77.90/93.88**	7.88/44.65M
ResNet-152	78.43/94.11	11.58/60.19M
IC-ResNet-152	**78.58/94.23**	11.62/60.34M
SE-ResNet-50	77.63/93.64	4.13/28.09M
IC-SE-ResNet-50	**78.02/93.88**	4.15/28.13M
SE-ResNet-101	78.35/94.09	7.86/49.33M
IC-SE-ResNet-101	**78.52/94.20**	7.89/49.42M
ResNeXt-101	79.30/94.54	16.51/88.79M
IC-ResNeXt-101	**79.52/94.78**	16.56/88.99M

ImageNet. We construct another set of DL experiments to evaluate some deeper IC networks on the ImageNet (Krizhevsky, Sutskever, & Hinton, 2012b) dataset. The ImageNet, a large image dataset widely used to evaluate the deep architectures, consists of more than 1 million color images in 1000 classes divided into 1.28M training images and 50K validation images. We apply the IC structure to some deep networks, including ResNet-50, ResNet-101, ResNet-152, SE-ResNet-50, SE-ResNet-101, and ResNeXt-101 ($32 \times 8d$). The optimizer uses the SGD method with a weight decay of 10^{-4} and a momentum of 0.9. The training process is set to 120 epochs with a batch size of 256. The learning rate is initially set to 0.1 and will be reduced 10 times every 30 epochs. Note that the reported results of basic models are reproduced with the same settings, which achieve or even exceed the results reported in the original papers.

From Table 11, we observe that the IC neuron can improve the performance of networks with different depths. More importantly, the additional FLOPs/Params of deep IC models can be negligible (<0.5%/0.3%). We believe that the IC neuron provides a more efficient way to improve the performance of deep models, compared to increasing the depth or width. Besides, it is well known that one disadvantage of deep learning is its complex hyperparameter design (Xie et al., 2017). Building the IC networks based on the existing models can bypass this obstacle. In summary, the DL experiments show that the advantage of the IC structure will not disappear as the depth of the networks increases. In addition, it provides a simpler way for model builders to build suitable deep models.

Object detection. The DL experiments focus on the image classification task. To further investigate the applicability of the IC structure, we evaluate the IC networks on the object detection. We use the PASCAL VOC 2007+2012 detection benchmark (Everingham, Gool, Williams, Winn, & Zisserman, 2010) whose dataset consists of about 5K train/val images and 5K test images over 20 object categories. We use the Faster R-CNN (Ren, He, Girshick, & Sun, 2015) and Retinanet (Lin, Goyal, Girshick, He, & Dollár, 2017) which are widely-used detection frameworks. To facilitate the application on object detection, we only replace the backbone networks with the corresponding IC networks. We use the same configuration described in Chen et al. (2019) and use the mean Average Precision (mAP) as the evaluation metric. As shown in Table 12, the IC-ResNet-50 exceeds ResNet-50 by 1.0% and 0.9% on the Faster R-CNN and Retinanet frameworks, respectively. The purpose of the backbone network is to extract the features of the objects in the image. Therefore, the object detection experiments imply that the IC networks can capture more fine-grained features to improve the overall performance.

Table 12
Results on PASCAL VOC 2007+2012 test set.

Framework	Backbone	mAP
Faster R-CNN	ResNet-50	79.5
	IC-ResNet-50	**80.1**
Retinanet	ResNet-50	77.3
	IC-ResNet-50	**78.0**

3.7. Discussion

In summary, a network composed of IC neurons can achieve a generalization performance better than that of MP neurons in a wide range of tasks. Experiments show that this improvement does not come from the increase in the number of calculations and parameters. On the one hand, compared to traditional networks that use the same hyperparameters, the additional computational burden brought by the IC network is basically negligible. On the other hand, we find it difficult for deeper and wider traditional networks to surpass the accuracy of IC networks. We think this is because the IC neuron has more non-linear representation capabilities and can fit more fine-grained features. In the training process, this advantage also speeds up the convergence speed. Besides, we integrate the IC neurons into modern deep CNN architectures. This set of experiments further show the applicability and universality of the IC neuron. We believe that the IC neuron can be combined with many existing networks to achieve good performance.

4. Conclusion

Inspired by the elastic collision model, we propose the IC neuron model that could be used to fit more complex distributions compared to the MP neuron. We build the IC networks by integrating the IC neurons into the popular network structures, including the FC, the convolutional and the recurrent structures. Experiments on a variety of classification tasks show that the IC networks can reach a higher accuracy and a faster convergence speed compared to the classical networks. Besides, the IC networks bring little additional computational burden, showing the superiority of the IC neuron. Through deep learning experiments, we further demonstrated the applicability of IC neurons in modern neural networks. Above all, our work provides a basic unit for building more efficient networks.

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.

Acknowledgment

This work is supported in part by the National Nature Science Foundation of China under Grant No. (61876076).

References

Attia, M. A., Sallam, E. A., & Fahmy, M. M. (2012). Single multiplicative neuron model based on generalized mean. In *2012 seventh international conference on computer engineering systems (icces)* (pp. 111–116).

Cazé, R. D., Jarvis, S., Foust, A. J., & Schultz, S. R. (2017). Dendrites enable a robust mechanism for neuronal stimulus selectivity. *Neural Computation*, *29*(9), 2511–2527.

Chen, K., Wang, J. Q., Pang, J. M., Cao, Y. H., Xiong, Y., Li, X. X., et al. (2019). Mmdetection: Open mmlab detection toolbox and benchmark. arXiv:1906.07155.

Clevert, D., Unterthiner, T., & Hochreiter, S. (2016). Fast and accurate deep network learning by exponential linear units (ELUs). In *4th international conference on learning representations, iclr 2016, san juan, puerto rico, may 2-4, 2016, conference track proceedings*.

Debanne, D., Campanac, E., Bialowas, A., Carlier, E., & Alcaraz, G. (2011). Axon physiology. *Physiological reviews*.

Egrioglu, E., Yolcu, U., Aladag, Ç. H., & Bas, E. (2015). Recurrent multiplicative neuron model artificial neural network for non-linear time series forecasting. *Neural Process. Lett.*, *41*(2), 249–258.

Eliasmith, C., Stewart, T. C., Choo, X., Bekolay, T., DeWolf, T., Tang, Y., et al. (2012). A Large-Scale Model of the Functioning Brain, Vol. 338, no. 6111, pp. 1202–1205.

Everingham, M., Gool, L. V., Williams, C. K. I., Winn, J. M., & Zisserman, A. (2010). The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, *88*(2), 303–338.

Gao, S., Zhou, M., Wang, Y., Cheng, J., Yachi, H., & Wang, J. (2019). Dendritic neuron model with effective learning algorithms for classification, approximation, and prediction. *IEEE Transactions on Neural Networks Learning Systems*, *30*(2), 601–614.

Graves, A., Mohamed, A.-r., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *2013 ieee international conference on acoustics, speech and signal processing* (pp. 6645–6649). IEEE.

He, K. M., Zhang, X. Y., Ren, S. Q., & Sun, J. (2016). Deep residual learning for image recognition. In *Cvpr* (pp. 770–778).

Howard, A. G., Zhu, M. l., Chen, B., Kalenichenko, D., Wang, W. J., Weyand, T., et al. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. In *Cvpr*.

Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. In *Cvpr* (pp. 7132–7141).

Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Cvpr*.

Hussain, A. J., Liatsis, P., Tawfik, H., Nagar, A. K., & Al-Jumeily, D. (2008). Physical time series prediction using recurrent pi-sigma neural networks. *International Journal of Artificial Intelligence Soft Computing*, *1*(1), 130–145.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd international conference on machine learning, icml 2015, lille, france, 6-11 july 2015, Vol. 37* (pp. 448–456).

Janacek, G. J. (2001). Practical time series.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012a). Imagenet classification with deep convolutional neural networks. In P. L. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems 25: 26th annual conference on neural information processing systems 2012. proceedings of a meeting held december 3-6, 2012, lake tahoe, nevada, united states* (pp. 1106–1114).

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012b). Imagenet classification with deep convolutional neural networks. In *Neurips* (pp. 1097–1105).

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84–90.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., et al. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, *1*(4), 541–551.

Li, C. (2003). A sigma pi sigma neural network (SPSNN). *Neural Processing Letters*, *17*(1), 1–19.

Lin, T., Goyal, P., Girshick, R. B., He, K. M., & Dollár, P. (2017). Focal loss for dense object detection. In *Iccv* (pp. 2999–3007).

Lyutikova, L. A. (2018). Sigma-pi neural networks: Error correction methods. In A. V. Samsonovich, & C. Lebiere (Eds.), *Procedia Computer Science*: *145*, *Postproceedings of the 9th Annual International Conference on Biologically Inspired Cognitive Architectures, BICA 2018 (Ninth Annual Meeting of the BICA Society), August 22-24, 2018, Prague, Czech Republic* (pp. 312–318). Elsevier.

Mackey, M., & Glass, L. (1977). Oscillation and chaos in physiological control systems. *Science*, *197*(4300), 287–289.

Mairal, J., Koniusz, P., Harchaoui, Z., & Schmid, C. (2014). Convolutional kernel networks. In *Advances in neural information processing systems* (pp. 2627–2635).

McCulloch, W. S., & Pitts, W. (1990). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biology*, *52*(1–2), 99–115.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Neural information processing systems* (pp. 3111–3119).

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel* (pp. 807–814).

Nayak, J., Naik, B., & Behera, H. (2014). A hybrid PSO-GA based pi sigma neural network (PSNN) with standard back propagation gradient descent learning for classification. In *2014 international conference on control, instrumentation, communication and computational technologies (iccicct)* (pp. 878–885).

Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). Activation functions: Comparison of trends in practice and research for deep learning. CoRR abs/1811.03378.

Ostojic, S., & Brunel, N. (2011). From spiking neuron models to linear-nonlinear models. *PLoS Computational Biology*, *7*(1).

Ren, S. Q., He, K. M., Girshick, R. B., & Sun, J. (2015). Faster r-CNN: towards real-time object detection with region proposal networks. In *Neurips* (pp. 91–99).

Sak, H., Senior, A. W., & Beaufays, F. (2014). Long short-term memory recurrent neural network architectures for large scale acoustic modeling.

Shin, Y., & Ghosh, J. (1991). The pi-sigma network: an efficient higher-order neural network for pattern classification and function approximation. *i*, In *IJCNN-91-Seattle International Joint Conference on Neural Networks* (pp. 13–18 vol.1).

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *Iclr*.

Todo, Y., Tang, Z., Todo, H., Ji, J., & Yamashita, K. (2019). Neurons with multiplicative interactions of nonlinear synapses. *International Journal of the Neural Systems*, *29*(8), 1950012:1–1950012:18.

Valueva, M. V., Nagornov, N., Lyakhov, P. A., Valuev, G. V., & Chervyakov, N. I. (2020). Application of the residue number system to reduce hardware costs of the convolutional neural network implementation. *Mathematics and Computers in Simulation*, *177*, 232–243.

Wang, C., Yang, J., Xie, L., & Yuan, J. (2019). Kervolutional neural networks. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 31–40).

Woźniak, S., Pantazi, A., Bohnstingl, T., & Eleftheriou, E. (2020). Deep learning incorporating biologically inspired neural dynamics and in-memory computing. *Natural Machine Intelligence*, *2*, 325–336.

Xie, S. N., Girshick, R., Dollár, P., Tu, Z. W., & He, K. M. (2017). Aggregated residual transformations for deep neural networks. In *Cvpr* (pp. 1492–1500).

Yadav, R. N., Kalra, P. K., & John, J. (2007). Time series prediction with single multiplicative neuron model. *Applied Soft Computing*, *7*(4), 1157–1163.

Zaremba, W., Sutskever, I., & Vinyals, O. (2014). Recurrent neural network regularization. CoRR abs/1409.2329.

Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2017). Understanding deep learning requires rethinking generalization. In *5th international conference on learning representations, iclr 2017, toulon, france, april, 2017, conference track proceedings*.

Zhou, Z., & Feng, J. (2017). Deep forest: Towards an alternative to deep neural networks. In *Twenty-sixth international joint conference on artificial intelligence* (pp. 3553–3559).