Hindawi
Computational Intelligence and Neuroscience
Volume 2023, Article ID 7037124, 13 pages
https://doi.org/10.1155/2023/7037124

Research Article

An Extension Network of Dendritic Neurons

Qianyi Peng,[1] **Shangce Gao**,[1] **Yirui Wang**,[2] **Junyan Yi**,[3] **Gang Yang**,[4] **and Yuki Todo**[5]

[1]*Faculty of Engineering, University of Toyama, Toyama-shi 930-8555, Japan*
[2]*Faculty of Electrical Engineering and Computer Science, Ningbo University, Ningbo, Zhejiang 315211, China*
[3]*Department of Computer Science & Technology, Beijing University of Civil Engineering and Architecture, Beijing 100044, China*
[4]*School of Information, Renmin University of China, Beijing, China*
[5]*Faculty of Electrical, Information and Communication Engineering, Kanazawa University, Kanazawa, Ishikawa 9201192, Japan*

Correspondence should be addressed to Shangce Gao; gaosc@eng.u-toyama.ac.jp

Received 15 October 2022; Revised 8 December 2022; Accepted 7 January 2023; Published 23 January 2023

Academic Editor: José Alfredo Hernández-Pérez

Deep learning (DL) has achieved breakthrough successes in various tasks, owing to its layer-by-layer information processing and sufficient model complexity. However, DL suffers from the issues of both redundant model complexity and low interpretability, which are mainly because of its oversimplified basic McCulloch–Pitts neuron unit. A widely recognized biologically plausible dendritic neuron model (DNM) has demonstrated its effectiveness in alleviating the aforementioned issues, but it can only solve binary classification tasks, which significantly limits its applicability. In this study, a novel extended network based on the dendritic structure is innovatively proposed, thereby enabling it to solve multiple-class classification problems. Also, for the first time, an efficient error-back-propagation learning algorithm is derived. In the extensive experimental results, the effectiveness and superiority of the proposed method in comparison with other nine state-of-the-art classifiers on ten datasets are demonstrated, including a real-world quality of web service application. The experimental results suggest that the proposed learning algorithm is competent and reliable in terms of classification performance and stability and has a notable advantage in small-scale disequilibrium data. Additionally, aspects of network structure constrained by scale are examined.

1. Introduction

In recent years, deep learning (DL) has dominated the research field of artificial intelligence (AI) and achieved dramatic successes in terms of speech recognition, protein structure prediction, drug discovery, image and video processing, and others. [1]. At present, the mainstream deep learning models are mostly constructed based upon neural networks, referring to multiple-layered parameterized McCulloch–Pitts neurons inspired by the biological neuron [2]. Neural networks as black boxes are not only extensively studied in the field of artificial intelligence but also highly applied to the industry of information technology [3, 4]. The appearance of deep neural networks pushes the development of neural networks to a new peak. However, given the numerous difficulties and problems they face, including the lack of a theoretical foundation for explanation [5, 6], fairness [7, 8], and causal discovery [9, 10], neural networks tend to become stuck in a relatively inert state. In pace with the increasing attention of interpretative theory, it is essential to urgently require newer and better discoveries and more valuable research orientations to avoid blindness for the promotion of scientific and technological progress.

From the perspective of understanding the mechanics of artificial neural networks, various methods and contributions are introduced [11, 12]. Through the application of the Monte Carlo simulation for quantifying the variable's importance, Olden et al. justified the rightness of the connection weight calculation method in neural networks [13]. Statistics pointed out that Sarle et al. proclaimed the relations between neural networks and the generalized linear model, maximum redundancy analysis, projection tracking, clustering analysis, and other statistical models and also transformed terms in neural networks into statistical terms

[14]. Similarly, certain relations between artificial neural networks and statistical methods were proposed in [15].

Despite putting forward the abovementioned studies, the bottleneck of neural networks is not addressed. Originating from the simulation of neurons in biological concepts, the artificial neural network takes artificial neurons as nodes to construct a complete conduction structure. As a basic information processing unit, the neuron is formed by a dendrite, cell body (soma), axon, and synapse, as shown in Figure 1(a). To be specific, the dendrite receives information from the outside world, the cell body processes the information, and the axon and synapse transmit the signal and pass it on to other neurons, respectively [16]. The structure of biological neurons can be traced back to the 1940s, when McCulloch and Pitts jointly published the abstract neuron model McCulloch–Pitts (MP) [17] for the first time, as illustrated in Figure 1(b). Then, in 1949, the Hebb rule [18] was proposed based on the theory of the variability of synapse connections of neurons within the human brain. The adjusting weight method was introduced into machine learning, thereby laying the foundation for the learning algorithms.

Inspired by the structure of biological neurons and MP, Todo et al. proposed the dendritic neural model (DNM) [19]. Different from the traditional neural network model, the dendritic neural model is designed based on neuron conduction and single neuron. DNM obtains the support of the biological theory to simulate the biological neuron. Furthermore, DNM compensates for some defects of the homologous perceptron model, such as the inability to solve the XOR problem. At the same time, the novel study of the human brain [20] is also brought about.

As a classifier, shown in Figure 1(c), DNM has been applied to various classification problems. For example, Sha et al. classified the breast cancer dataset, and Jiang et al. detected the liver disorder [21] for assisted disease diagnosis. Apart from the development of medical aid applications, an unconventional method was also applied to the financial field. To improve the classification performance, metaheuristic algorithms were introduced to train the hyperparameters of DNM [22, 23]. Through the use of the decision tree, Luo et al. initialized the model to realize better effectiveness [24]. For solving a generalized large-scale classification problem, Jia et al. suggested a reconciliation method with DNM by using a particle antagonism mechanism, and Ji et al. proposed a DNM-based multiobjective evolutionary algorithm [25]. In terms of feature selection, Song et al. addressed the high-dimensional challenge [26], and Gao et al. also showed the expansibility and flexibility of DNM for diverse applications [27]. Utilizing the multiplication operation that is useful to the information processing for a single neuron, the computing in synapses is imaginatively described using sigmoid functions. It is advantageous to establish the morphology of a neuron by determining the values of the parameters in synapses since the output of synapses can effectively represent signals. Nevertheless, it is noted that the single neuron is limited in partial application scenarios. In [28], the binary classification results of DNM were incorporated to undertake multiple classification tasks and thus recognize the multiclassification datasets.

By adopting the quality of service (QoS) as the evaluation dataset, this study implements the multiclassification of web service selection. QoS is defined as the fact that a network utilizes a range of basic technologies to provide superior service capabilities for the designated network communication [29]. As a security mechanism of the network and a technology, QoS is carried out to deal with network delay [30], blocking [31], and other problems. For a general situation, the common network bandwidth as a significant metric is instanced in order to illustrate QoS. When the standard of service quality has not appeared, the network environment treats all services and applications in an equal way, resulting in a disordered situation, as shown in Figure 2(a) where the colored area stands for different web services and applications. In other words, when a network device does not have the capacity of QoS, the network environment will be threatened, and a bottleneck will be created [32]. As shown in Figure 2(b), prioritization from the perspective of QoS provides a more orderly, efficient, and stable network environment. QoS contains a set of nonfunctional attributes, which is the measure and criteria of such characteristics of the web services, such as reliability and response time, to effectively classify and sort different services.

Web services refer to some software modules running on the network, which are service-oriented and based on distributed programs. Due to the fact that the web service employs general Internet standards, such as HTTP and XML (a subset of the standard generalized markup language) [33], human beings then have access to data on the web via various terminal devices in various places. In this article, the described web service is different from the common network application. It generally refers to some application modules, such as the network protocol and method, which is the basis of network applications. With the development of the Internet, many candidate services have implemented the same task, and most of them have the same functions but different nonfunctional characteristics. As a result, these services are divided into different service quality levels. Overlapping is seen to be inevitable because of the existence of a wide range of web services on the network. Based on the QoS, web service selection is considered an effective solution [34]. As network technology and operation concepts develop rapidly, web services are becoming the latest technology and development trend for constructing distributed, modular, and service-oriented applications.

Based on DNM, this article proposes a multiple dendritic neural network (MDNN) with multiple single neurons to achieve the multiclassification of web service selection based on QoS. To adjust the multiclassification mechanism, the structure of DNM introduced in Figure 3 is reconstructed. For the purpose of accelerating the gradient descent and improving the multiclassification accuracy, the back-propagating algorithm and adaptive moment estimation optimization are derived for the first time. Experiments are carried out on the Quality of Web Service dataset and nine UCI multiclassification datasets [35]. In the comparison between MDNN and nine state-of-the-art classifiers, the superiority of the proposed method is demonstrated.

The contributions are majorly classified into the aspects as follows: 1) a novel multiple single-neuron neural network

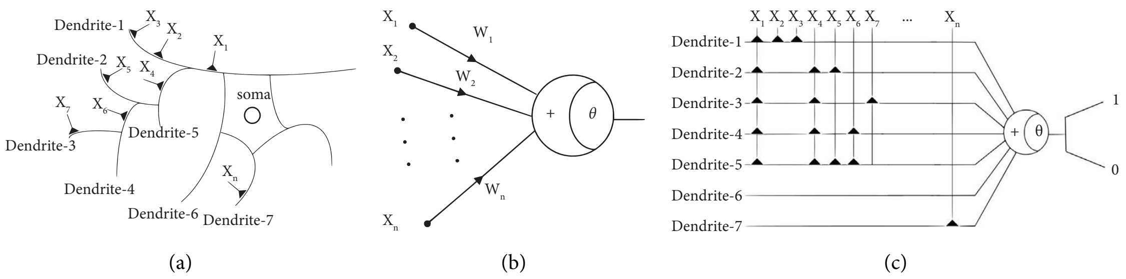

Figure 1: (a) The function of a biological neuron is completely different depending on the shape of its dendrites and the location of its synapses. (b) McCulloch–Pitts neuron model: no interaction in dendrite morphology and dendrites. (c) Single dendritic neural model: faithful representation of dendrite morphology and dendrites fixated on binary classification.

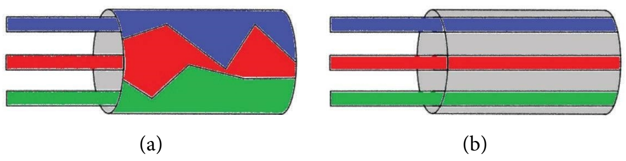

Figure 2: (a) The network is disordered without QoS rules. (b) The network is in order with QoS rules.

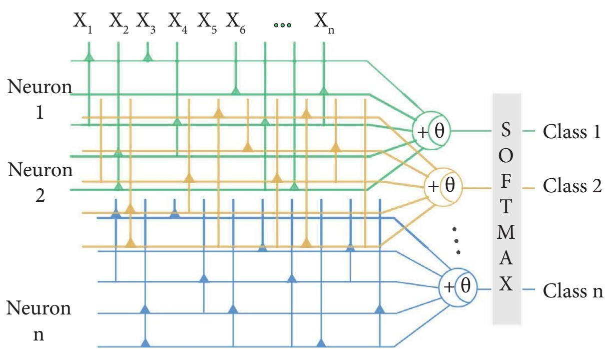

Figure 3: Multiple dendritic neural networks: applied on comprehensive applications based on the dendritic structure.

for multiclassification tasks is developed. 2) The potential and application scenarios of the dendritic neural network are explored. 3) A new approach for QoS-driven web service selection is proposed.

Given as follows is the organization of the remaining parts of this article: Section 2 presents the structure of the multiple dendritic neural networks. Section 3 elaborates on the learning processes of the proposed method and expounds on the optimization strategies. The comparison with other algorithms and experimental results are shown in Section 4. At last, Section 5 concludes the paper and formulates future work.

2. The Dendritic Neuron Network-Based Multiclassifier Approach

The proposed multiclassifier is constructed by multiple single neurons. The general architecture is shown in Figure 4. As for each neuron, x_i, the input of the model is preprocessed by using a nonlinear sigmoid filter. To differentiate neurons, the function introduces the subscript j, which is defined as follows:

$$Y_{j,i,m} = \frac{1}{1 + e^{-\left(w_{j,i,m}x_i - q_{j,i,m}\right)}}, \tag{1}$$

where i is the number of attributes of the sample, m is the number of nodes within the hidden layer, and j is the number of classifications of output results. In addition, the weight $w_{j,i,m}$ and threshold $q_{j,i,m}$ denote the neural network parameters in the training stage and are randomly initialized within (0, 0.01) and 0, respectively.

In contrast to the perceptron model, a quadrature method is adopted for the hidden layer to not only rule out the inhibited neuronal excitation but also enhance the activated neuronal excitation. The formula is described as follows:

$$Z_{j,m} = \prod_{i=1} Y_{j,i,m}, \tag{2}$$

$$V_j = \sum_{m=1} Z_{j,m}. \tag{3}$$

Eq. (2) means that all of the hidden layers are activated, which is equivalent to a logical AND. Eq. (3) is equivalent to a logical OR where all inhibited neuronal excitations from the former layer are suppressed exclusively and the rest are reactivated. As a result, the multiclassification structure of multiple neurons is formed.

Apart from the dendritic mechanism, MDNN utilizes the normalized exponential function to output final results. For ease of consistency in representation, the illustrated style of the normalized exponential function is followed. To be noted, the output of multiple neurons is processed by all information from the previous layer instead of being directly conveyed, expressed as follows:

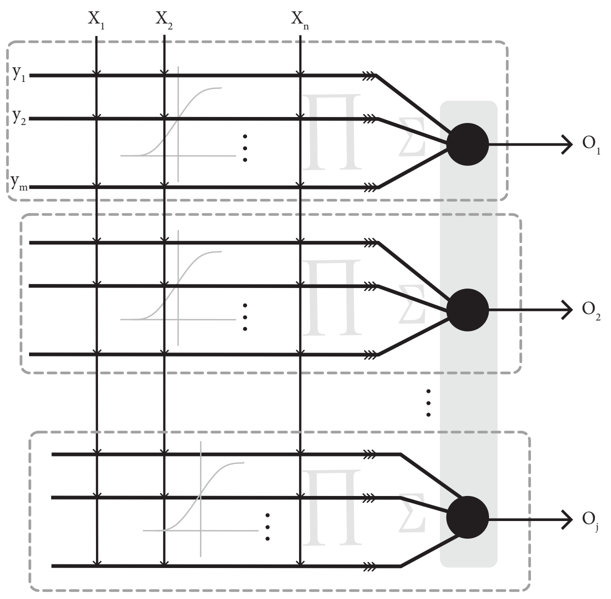

Figure 4: The structure of the proposed MDNN. The framed rectangle represents the structure of a single neuron.

$$O_j = \frac{e^{V_j}}{\sum_{j=1} e^{V_j}}, \tag{4}$$

where O_j is the possibility of the prediction for each class. The normalized exponential function converts the output value of the upper layer, V_j, to the probability distribution with the range of [0, 1], and the sum of the probability values of each neuron being 1. The formula first converts the results V_j into an exponential function, ensuring the nonnegative probability, and then normalizes the probability values into 1.

Since the prediction result follows the rules of the probability distribution, the cross-entropy function as the loss function is considered a proper substitute for mean square error, which is defined as follows:

$$E_j = -\sum_{j=1} T_j * \log O_j, \tag{5}$$

where E_j represents the similarity of probability distribution between the prediction of the model and the actual classification and T_j is the actual classification label.

3. Learning Mechanism and Optimization Strategies

The existing learning algorithms cannot be directly applied since MDNN is a new dendritic neuron model containing multiplication operators in its calculation. Accordingly, in this section, we for the first time derive the learning algorithms for our proposed MDNN, specifically one is the traditional error backpropagation, and the other is an Adam-like learning algorithm.

3.1. Backpropagation. In the course of learning samples, the model is promoted by the stochastic gradient descent of parameters $w_{j,i,m}$ and $q_{j,i,m}$, which is described as follows:

$$\begin{aligned} w^t_{j,i,m} &= w^{t-1}_{j,i,m} - \eta \Delta w_{j,i,m}, \\ q^t_{j,i,m} &= q^{t-1}_{j,i,m} - \eta \Delta q_{j,i,m}, \end{aligned} \tag{6}$$

where η as the learning rate is a positive constant. t and $t-1$ denote the current iteration and the previous iteration in the training stage, respectively.

The error of the proposed MDNN is calculated by the cross-entropy function. According to the calculated error, the error backpropagation algorithm is introduced as the learning scheme. In backpropagation, all the samples or a batch of samples are involved. To better realize intuition, the relation among layers is shown in Figure 5.

$\Delta w_{j,i,m}$ and $\Delta q_{j,i,m}$ are expressed by the partial differential form as follows:

$$\begin{aligned} \Delta w_{j,i,m} &= \frac{\partial E_j}{\partial O_j}\frac{\partial O_j}{\partial V_j}\frac{\partial V_j}{\partial Z_{j,m}}\frac{\partial Z_{j,m}}{\partial Y_{j,i,m}}\frac{\partial Y_{j,i,m}}{\partial w_{j,i,m}}, \\ \Delta q_{j,i,m} &= \frac{\partial E_j}{\partial O_j}\frac{\partial O_j}{\partial V_j}\frac{\partial V_j}{\partial Z_{j,m}}\frac{\partial Z_{j,m}}{\partial Y_{j,i,m}}\frac{\partial Y_{j,i,m}}{\partial q_{j,i,m}}. \end{aligned} \tag{7}$$

Since the model is trained by batches, $\Delta w_{j,i,m}$ and $\Delta q_{j,i,m}$ obtained by the gradient descent are finally calculated as follows:

$$\begin{aligned} \overline{\Delta w_{j,i,m}} &= \frac{\Delta w_{j,i,m}}{N}, \\ \overline{\Delta q_{j,i,m}} &= \frac{\Delta q_{j,i,m}}{N}, \end{aligned} \tag{8}$$

where N denotes the size of input data within the current iteration.

Following the chain rule, the derivation procedures and results are presented according to the backpropagation. Firstly, the partial differential of error E is calculated. By the empirical evidence of normalized exponential function,

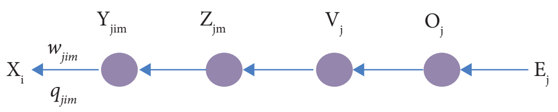

Figure 5: The relation among layers.

$\partial E_j/\partial O_j$ and $\partial O_j/\partial V_j$ are computed collectively instead of computing them separately.

The forward propagation for the multiclassification is not directly corresponding. Thus, the derivation of error E is discussed in Cases (1) and (2). To avoid confusion, the subscripts of V and O are redefined as m and n, respectively. Their relation is simplified as follows:

$$V_m \longleftarrow O_n \longleftarrow E_n. \tag{9}$$

On the basis of Equations. (5) and (4), when $n = m$, there is Case (1):

$$\begin{aligned}\frac{\partial E_n}{\partial V_m} &= \frac{\partial E_n}{\partial O_n}\frac{\partial O_n}{\partial V_m},\\ &= -T_n\frac{1}{O_n}\left[O_n\left(1-O_n\right)\right],\\ &= -T_n\left(1-O_n\right).\end{aligned} \tag{10}$$

When $n \neq m$, there is Case (2):

$$\begin{aligned}\frac{\partial E_n}{\partial V_m} &= -\sum_{n\neq m} T_m\frac{1}{O_m}\left(-O_nO_m\right)\\ &= \sum_{n\neq m} T_mO_n.\end{aligned} \tag{11}$$

We incorporate Cases (1) and (2) into the following formula:

$$\begin{aligned}\frac{\partial E_n}{\partial V_m} &= -T_n\left(1-O_n\right) + \sum_{n\neq m} T_mO_n,\\ &= -T_n + T_nO_n + \sum_{n\neq m} T_mO_n,\\ &= O_n\left(T_n + \sum_{n\neq m} T_m\right) - T_n,\\ &= O_n - T_n.\end{aligned} \tag{12}$$

Thus, $\partial E_j/\partial V_j$ is expressed as follows:

$$\frac{\partial E_j}{\partial V_j} = \frac{\partial E_j}{\partial O_j}\frac{\partial O_j}{\partial V_j} = O_j - T_j. \tag{13}$$

For the rest layers of MDNN, they are derived according to Equations. (3) and (2) as follows:

$$\frac{\partial V_j}{\partial Z_{j,m}} = 1, \tag{14}$$

$$\frac{\partial Z_{j,m}}{\partial Y_{j,i,m}} = \frac{Z_{j,m}}{Y_{j,i,m}}. \tag{15}$$

Taking the derivative of Equation. (1) with the sigmoid function, $\partial Y_{j,i,m}/\partial w_{j,i,m}$ and $\partial Y_{j,i,m}/\partial q_{j,i,m}$ are obtained as follows:

$$\begin{aligned}\frac{\partial Y_{j,i,m}}{\partial w_{j,i,m}} &= x_i \cdot Y_{j,i,m}\left(1-Y_{j,i,m}\right),\\ \frac{\partial Y_{j,i,m}}{\partial q_{j,i,m}} &= -Y_{j,i,m}\left(1-Y_{j,i,m}\right).\end{aligned} \tag{16}$$

3.2. Adam-Like Optimization. For improving the convergence and classification ability of the proposed model, inspired by the well-known adaptive moment estimation (Adam) [36], an Adam-like learning algorithm for MDNN is also introduced to accelerate the gradient descent without diverging. The way of updating weights in each iteration is optional. The traditional way mentioned in Section 3.1 or Adam can be altered according to the user's setting.

As an extended optimization strategy of stochastic gradient descent (SGD) [37], momentum [38, 39] is introduced to reduce the oscillation and accelerate the gradient descent. The fundamental concept of gradient descent with momentum lies in updating the weight by calculating the exponentially weighted average of the gradient as follows:

$$v_{\Delta w_{j,i,m}} = \alpha v_{\Delta w_{j,i,m}} + (1-\alpha)\Delta w_{j,i,m}, \tag{17}$$

$$v_{\Delta q_{j,i,m}} = \alpha v_{\Delta q_{j,i,m}} + (1-\alpha)\Delta q_{j,i,m}, \tag{18}$$

where α is a positive constant to smooth out the gradient descent process. Intuitively, $\Delta w_{j,i,m}$ and $\Delta q_{j,i,m}$ are interpreted as the acceleration in physics. $v_{\Delta w_{j,i,m}}$ and $v_{\Delta q_{j,i,m}}$ are regarded as the velocity, and α is seen as the friction. In addition, $\Delta w_{j,i,m}$ and $\Delta q_{j,i,m}$ accelerate the gradient descent and gain the velocity $\Delta w_{j,i,m}$ and $\Delta q_{j,i,m}$, and the friction α prevents the acceleration.

In this case, the updates of parameters $w_{j,i,m}$ and $q_{j,i,m}$ are modified as follows:

$$w_{j,i,m}^{t} = w_{j,i,m}^{t-1} - \eta v_{\Delta w_{j,i,m}}, \tag{19}$$

$$q_{j,i,m}^{t} = q_{j,i,m}^{t-1} - \eta v_{\Delta q_{j,i,m}}. \tag{20}$$

Serving as a crucial part of Adam, the root mean square prop (RMSprop) [40] auxiliary accelerates the gradient descent as follows:

$$u_{\Delta w_{j,i,m}} = \beta v_{\Delta w_{j,i,m}} + (1-\beta)\Delta w_{j,i,m}^{2}, \tag{21}$$

where β is a positive constant similar to α. $w_{j,i,m}$ is calculated as follows:

$$w_{j,i,m}^{t} = w_{j,i,m}^{t-1} - \eta \frac{\Delta w_{j,i,m}}{\sqrt{v_{\Delta w_{j,i,m}}}}. \tag{22}$$

Similarly, $u_{\Delta q_{j,i,m}}$ is obtained by

$$u_{\Delta q_{j,i,m}} = \beta v_{\Delta q_{j,i,m}} + (1-\beta)\Delta q_{j,i,m}^{2}, \tag{23}$$

$$q_{j,i,m}^{t} = q_{j,i,m}^{t-1} - \eta \frac{\Delta q_{j,i,m}}{\sqrt{v_{\Delta q_{j,i,m}}}}. \tag{24}$$

In order to avoid the bias of exponentially weighted average in the initial learning stage, Equations. (17), (18), (21), and (23) are modified to obtain more accurate results as follows:

$$\begin{aligned} v_{\mathrm{corr}}^{\Delta w_{j,i,m}} &= \frac{v_{\Delta w_{j,i,m}}}{1-\alpha^{t}}, \\ v_{\mathrm{corr}}^{\Delta q_{j,i,m}} &= \frac{v_{\Delta q_{j,i,m}}}{1-\alpha^{t}}, \\ u_{\mathrm{corr}}^{\Delta w_{j,i,m}} &= \frac{u_{\Delta w_{j,i,m}}}{1-\beta^{t}}, \\ u_{\mathrm{corr}}^{\Delta q_{j,i,m}} &= \frac{u_{\Delta q_{j,i,m}}}{1-\beta^{t}}. \end{aligned} \tag{25}$$

For the acceleration of the gradient descent, Adam combines RMSprop with momentum. Thus, based on Equations. (23) and (24), the parameter updating equations optimized by Adam are expressed as follows:

$$\begin{aligned} w_{j,i,m}^{t} &= w_{j,i,m}^{t-1} - \eta \left(\frac{v_{\mathrm{corr}}^{\Delta w_{j,i,m}}}{\sqrt{u_{\mathrm{corr}}^{\Delta w_{j,i,m}}} + \varepsilon} \right), \\ q_{j,i,m}^{t} &= q_{j,i,m}^{t-1} - \eta \left(\frac{v_{\mathrm{corr}}^{\Delta q_{j,i,m}}}{\sqrt{u_{\mathrm{corr}}^{\Delta q_{j,i,m}}} + \varepsilon} \right), \end{aligned} \tag{26}$$

where ε is an infinitesimal so as to prevent the computation overflow.

4. Experimental Evaluation

4.1. Experimental Setup. The quality of web service (QWS) dataset [41, 42] is a real-world dataset based on the quality of service. Several versions of QWS are available. In this study, experiments use the original version, which consists of 364 web services. Its quality is described by a total of 10 nonfunctional attribute indexes. The QWS dataset divides the web service into 4 levels from the highest to the lowest, which are platinum, gold, silver, and bronze.

To avoid overfitting while improving the accuracy of results, data preprocessing strategies are adopted in the experiments. The raw data are normalized by using the rule of standardization:

$$x = \frac{x - \overline{x}}{\sigma}. \tag{27}$$

Moreover, the normalized data are randomly divided into three parts: 70 percent for the training process, 15 percent for the testing process, and the remaining data for the validation, to reduce unnecessary time consumption.

4.2. Optimal Parameter Settings. All of the hyperparameters are illustrated in Table 1, which presents their description and values. Sigmoid, tanh, Rectified Linear Unit (ReLU), and Leaky ReLU are available to freely choose the suitable active function. For large datasets, the samples are divided into mini-batch and shuffled for the gradient descent during the training stage. If batchsize is equal to the number of samples during the iteration, then the batch gradient descent will be executed.

However, it is tricky to determine the epoch size. To enable the model to reach optimal performance, a self-adaptive appending training epoch is arranged in the training stage according to the convergence of the validation process. Beginning with the default configuration, the epoch then adaptively increases pivoting on whether the gradient descent is approaching stagnation. At the same time, the initial value is set to 100 to avoid a higher epoch causing the time consumption. For general neural networks, experimental results are highly affected by the combination of parameters. Therefore, parameters, which are batchsize, M, η, and precision, are adjusted by the orthogonal experiment with 4 factors and 3 levels. The specific design is listed in Table 2. The optimal parameters of MDNN for QWS are finally shown in Table 3. The other parameters comply with the setting in Table 1.

4.3. Performance and Discussion. This section is divided into two subsections: experimental results of QWS analyzed by a variety of evaluation indicators are elaborated in Section 4.3.1, and Section 4.3.2 compares the proposed model with other multiclassification methods to demonstrate the prominent superiority of the proposed model.

4.3.1. Experimental Results. For the comprehensive performance evaluation of MDNN, the following statistical indicators are used: precision, recall, F_1 score, accuracy, and area under the curve (AUC) [43]. It is worth noting that the precision refers to the classification precision. For not only the convenience of the intuitive evaluation but also the justification of the following comparison, the macroaverage, which is the arithmetic average of performance indicators of all categories instead of instances, is adopted to statistically process classification results.

Table 4 shows the classification results of MDNN on the 4-class QWS dataset. The average values and optimum values represent the classification performance of MDNN. Although the dataset is technically unbalanced, such as the number of classes of QWS in Table 5, the stability and generalization ability of MDNN are considered to be effectively validated.

The five statistical indicators suggest that the classification achieved by MDNN for each class is effective, stable, and reliable. In their mean values, it is indicated that MDNN has good classification performance. The gradient descent

Table 1: Hyperparameters in the experiments.

ID	Parameters	Description	Default
1	activation	Activation function	Sigmoid
2	k	Parameter for Leaky ReLU	0.01
3	time	Operation times	30
4	η	Learning rate	0.01
5	M	Number of nodes for the hidden layer	5
6	epoch	Training epoch	100
7	batchsize	Mini-batch size for training	100
8	precision	End the training if reaching the precision	0.1
9	Ada m	Active or inactive the optimization	1
10	α	Parameter for Adam	0.9
11	β	Parameter for Adam	0.999
12	epsilon	Parameter for Adam	1e-8

Table 2: Orthogonal experimental design of hyperparameters with $L_9(3^4)$ for QWS.

No.	batchsize	M	precision	η
1	120	4	0.01	0.005
2	120	5	0.02	0.008
3	120	6	0.05	0.01
4	240	4	0.02	0.01
5	240	5	0.05	0.005
6	240	6	0.01	0.008
7	364	4	0.05	0.008
8	364	5	0.01	0.01
9	364	6	0.02	0.005

Table 3: Optimal parameter settings for QWS.

ID	Parameters	Optimal	Adaptive/Not adaptive
1	batchsize	364	Not adaptive
2	precision	0.01	Not adaptive
3	M	5	Not adaptive
4	η	0.01	Not adaptive
5	epoch	100	Adaptive

optimization strategy effectively reduces the errors. Moreover, MDNN accelerates the gradient descent to maintain a continuous downward trend, thereby finally guaranteeing the generalization and robustness of MDNN.

4.3.2. Comparison of Methods. To further verify the efficiency of MDNN, nine classifiers in total are used to compare with MDNN on nine multiclassification datasets from the UCI machine learning repository and the QWS dataset. The information of datasets and parameter settings of MDNN are listed in Table 5. The nine classifiers consist of BP, SVM, KNN, CART, naïve Bayes, LDA, QDA, J48, and random forest [44]. In addition, the ten datasets include Iris, Wine, Vehicle, Balance scale, CMC, Seed, Vowel, Thyroid, Robot navigation, and QWS.

Experimental results are shown in Table 6 where the best result for each dataset among all compared methods is highlighted in bold. According to five statistical indicators, it can be found that MDNN has the most optimal values in comparison with other classifiers. On the Iris, Wine, Seed, and Thyroid datasets, MDNN gives the best performance, with perfect outcomes of 100 percent correctness. Also, MDNN performs well with unbalanced data such as Vowel, Thyroid, and QWS. As a result, MDNN's classification performance is more constant than that of other methods. Nevertheless, MDNN appears to have a minor disadvantage on the large datasets, such as Balance scale, Vehicle, CMC, and Robot navigation, which seem to be constrained by the distribution of the network structure. In the comparison between MDNN and other classifiers, the superiority and effectiveness of the multiple dendritic neuron structure are verified.

The receiver operating characteristic (ROC) curves of ten multiclassification methods show the correct classification coverage of each class of the QWS dataset in Figure 6. It can be found that MDNN not only has a consistent performance on each class of QWS but also outperforms the other classifiers, thus indicating the effectiveness and stability of MDNN for the QWS classification and unbalanced multiclassification applications. Besides, experiments also demonstrate the efficiency and superiority of MDNN in terms of classification performance and stability.

4.4. Morphology and Logical Circle Realization. For the display of a data sample, the shuffle operation set in the pretraining period is at disposal. According to the initialization of $w_{j,i,m}$ and $q_{j,i,m}$ within Section 2, the synapses were calculated around 0.5 in the previous state. Through the

Table 4: Average and optimum values of classification results of MDNN for QWS.

		Platinum	Gold	Silver	Bronze	Mean values
Precision	(Average)	99.44	98.48	98.92	99.55	99.10
	(Optimum)	100.00	100.00	100.00	100.00	100.00
Recall	(Average)	96.52	99.72	99.46	98.81	98.63
	(Optimum)	100.00	100.00	100.00	100.00	100.00
F1	(Average)	97.73	99.05	99.16	99.15	98.77
	(Optimum)	100.00	100.00	100.00	100.00	100.00
Accuracy	(Average)	99.58	99.52	99.45	99.52	99.52
	(Optimum)	100.00	100.00	100.00	100.00	100.00
AUC	(Average)	99.11	99.71	99.92	99.97	99.68
	(Optimum)	100.00	100.00	99.85	100.00	99.96

Table 5: Description of datasets and parameter settings of MDNN.

Datasets	Instances	Number of features	Number of classes	Size of classes	M	η
Iris	150	4	3	50, 50, 50	4	0.01
Wine	178	13	3	59, 71, 48	5	0.01
Vehicle	846	18	4	199, 217, 218, 212	10	0.02
Balance scale	625	4	3	49, 288, 288	12	0.01
CMC	1473	9	3	629, 333, 511	20	0.03
Seed	210	7	3	70, 70, 70	5	0.01
Vowel	871	3	6	72, 89, 172, 151, 207, 180	6	0.01
Thyroid	215	5	3	150, 35, 30	4	0.01
Robot navigation	5456	24	4	82, 620, 972, 205, 329	30	0.02
QWS	364	10	4	41, 100, 120, 103	5	0.01

training stage, the weights $w_{j,i,m}$ and $q_{j,i,m}$ were gradually stabilized. As shown in Figure 7, synaptic changes, thus, yield to accomplish the pruning of the redundant network structure.

It can be easily observed that neuron 1, neuron 2, and neuron 4 are fully inhibited in accordance with the rule of Equation. (2) and Equation. (3). Consequently, the structure of neuron 1, neuron 3, and neuron 4 is ruled out. To specify the states of dendrites, a total of four scenarios are listed as follows:

$$\text{dendritic state} = \begin{cases} \text{constant} - 1\ \text{connection}, & \text{when } q_{j,i,m} < w_{j,i,m} < 0 \text{ or } q_{j,i,m} < 0 < w_{j,i,m}, \\ \text{constant} - 0\ \text{connection}, & \text{when } 0 < w_{j,i,m} < q_{j,i,m} \text{ or } w_{j,i,m} < 0 < q_{j,i,m}, \\ \text{excitatory connection}, & \text{when } 0 < q_{j,i,m} < w_{j,i,m}, \\ \text{inhibitory connection}. & \text{when } w_{j,i,m} < q_{j,i,m} < 0. \end{cases} \tag{28}$$

For the remainder neuron, the residual dendrite morphology is formed in Figure 8(a), which places the line of dashes indicating pruning. As mentioned previously in Section 2, logic-based inherent relations have existed within dendritic structures. Finally, since constant-1 holds no substantive impact on the attributes, the connections among dendrites are equivalent to logic OR. Thus, the hardware realization is transformed as illustrated in Figure 8(b), where the multiplexer is a 1 : 2 numerical compactor. In addition to showing the extendibility of MDNN, the overhead also indicates that MDNN can avoid overfitting by increasing the dendritic matrix sparsity.

Table 6: Comparison between MDNN and nine classifiers.

	Precision	Recall	F1	Accuracy	AUC	Precision	Recall	F1	Accuracy	AUC	Precision	Recall	F1	Accuracy	AUC
			BP					*SVM*					*KNN*		
Iris	96.30	96.13	95.91	97.39	99.68	96.43	96.11	96.06	97.39	99.76	96.46	96.52	96.29	97.68	98.78
Wine	27.74	43.30	31.65	65.60	62.29	43.62	40.25	31.31	64.28	72.29	71.02	71.02	69.10	80.74	87.78
Vehicle	12.26	28.46	14.91	63.57	54.03	12.11	24.65	10.29	61.77	38.12	62.86	64.22	62.19	82.18	86.18
Balance scale	67.50	67.84	66.37	93.33	90.65	65.70	66.60	65.36	91.35	94.59	59.70	56.52	57.24	83.43	65.29
CMC	37.23	42.85	38.26	65.69	64.15	54.45	51.64	51.69	70.37	72.06	47.22	46.01	45.06	66.06	65.75
Seed	90.79	90.43	89.89	93.54	97.56	90.39	90.24	89.98	93.47	98.82	89.05	88.72	88.02	92.43	96.36
Vowel	3.82	16.67	6.21	74.31	41.99	83.78	53.21	57.54	86.21	80.74	83.62	83.27	82.88	94.82	95.55
Thyroid	73.84	71.61	70.86	90.56	87.97	68.72	74.57	67.03	89.79	96.47	94.36	84.25	87.69	95.28	96.81
Robot navigation	73.76	63.39	65.32	88.18	91.38	90.18	87.52	88.73	94.76	98.17	85.03	84.86	84.73	92.68	95.79
QWS	17.64	30.71	19.42	68.94	51.34	85.03	84.86	84.73	92.68	32.69	62.16	60.49	59.39	78.61	81.92
			CART					*Naive Bayes*					*LDA*		
Iris	96.26	96.16	96.02	97.20	97.82	95.97	95.87	95.52	97.49	99.52	98.07	98.13	97.94	98.65	99.07
Wine	91.54	92.54	91.36	94.57	95.15	96.35	96.95	96.34	97.70	99.89	97.89	98.20	97.89	98.68	98.55
Vehicle	68.91	69.18	68.73	84.46	84.26	51.67	46.11	41.68	72.66	77.41	77.63	78.53	77.71	89.23	87.98
Balance scale	55.75	57.27	56.25	85.72	76.35	60.99	65.62	62.88	93.22	89.66	73.68	80.32	67.58	82.65	82.20
CMC	49.79	49.23	49.15	67.63	67.37	49.45	49.88	47.56	65.12	67.56	49.88	50.52	49.06	66.44	63.64
Seed	88.81	88.56	87.99	92.29	92.98	91.42	91.41	91.14	94.24	98.67	96.77	96.76	96.65	97.92	97.59
Vowel	96.77	96.76	96.65	97.92	97.59	77.72	77.50	76.89	93.22	96.75	75.36	76.85	75.06	92.62	84.00
Thyroid	87.46	88.22	86.31	94.58	93.00	97.34	95.77	96.15	98.26	99.65	95.81	88.15	90.53	95.69	93.08
Robot navigation	**99.17**	99.22	**99.19**	**99.67**	99.59	54.97	66.44	54.20	76.36	83.66	56.80	66.73	58.53	80.82	75.68
QWS	98.18	98.71	98.40	**99.73**	98.64	81.61	84.12	81.71	90.64	95.78	86.35	86.35	85.77	93.42	89.55
			QDA					*J48*					*Random Forest*		
Iris	97.60	97.61	97.44	98.36	98.82	100.00	97.78	98.83	94.07	98.89	**100.00**	**100.00**	**100.00**	94.81	**100.00**
Wine	98.95	98.60	98.67	99.18	99.18	92.54	93.19	92.63	91.57	94.68	97.15	98.12	97.56	96.26	99.82
Vehicle	85.20	85.49	85.02	92.57	92.19	86.67	89.89	88.15	71.60	94.06	**87.47**	**95.21**	**91.10**	74.88	**99.09**
Balance scale	**83.57**	**91.04**	**85.62**	94.47	91.71	3.82	2.71	3.03	77.92	53.65	2.75	3.40	3.02	80.33	53.38
CMC	**62.28**	60.57	60.86	**74.96**	**78.12**	61.36	61.25	61.20	51.94	69.31	59.32	**64.59**	**61.77**	50.67	71.65
Seed	94.94	94.40	94.48	96.39	95.22	86.98	86.83	86.59	91.06	89.59	87.91	90.32	88.92	92.49	96.11
Vowel	75.47	77.50	75.50	92.80	84.42	63.17	57.64	59.30	83.52	86.94	64.81	64.99	64.23	85.11	89.00
Thyroid	94.20	95.29	94.17	97.29	97.17	94.39	95.11	94.67	92.51	91.31	95.43	98.37	96.84	95.50	98.61
Robot navigation	64.42	75.59	66.73	83.44	82.74	98.90	99.37	99.13	99.40	99.67	98.61	**99.42**	99.01	99.41	**99.98**
QWS	83.65	85.18	83.70	91.52	89.31	97.99	**100.00**	**98.95**	99.76	99.86	97.04	96.75	96.78	98.99	**99.93**
			MDNN												
Iris	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**										
Wine	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**										
Vehicle	85.34	85.48	85.33	**93.31**	96.12										
Balance scale	80.50	77.76	78.92	**95.74**	**93.26**										
CMC	**62.28**	60.57	60.86	**74.96**	**78.12**										
Seed	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**										
Vowel	**91.29**	**84.26**	**86.47**	**95.67**	**98.03**										
Thyroid	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**										
Robot navigation	94.89	92.11	93.41	97.19	99.30										
QWS	**99.10**	98.63	98.77	99.52	99.68										

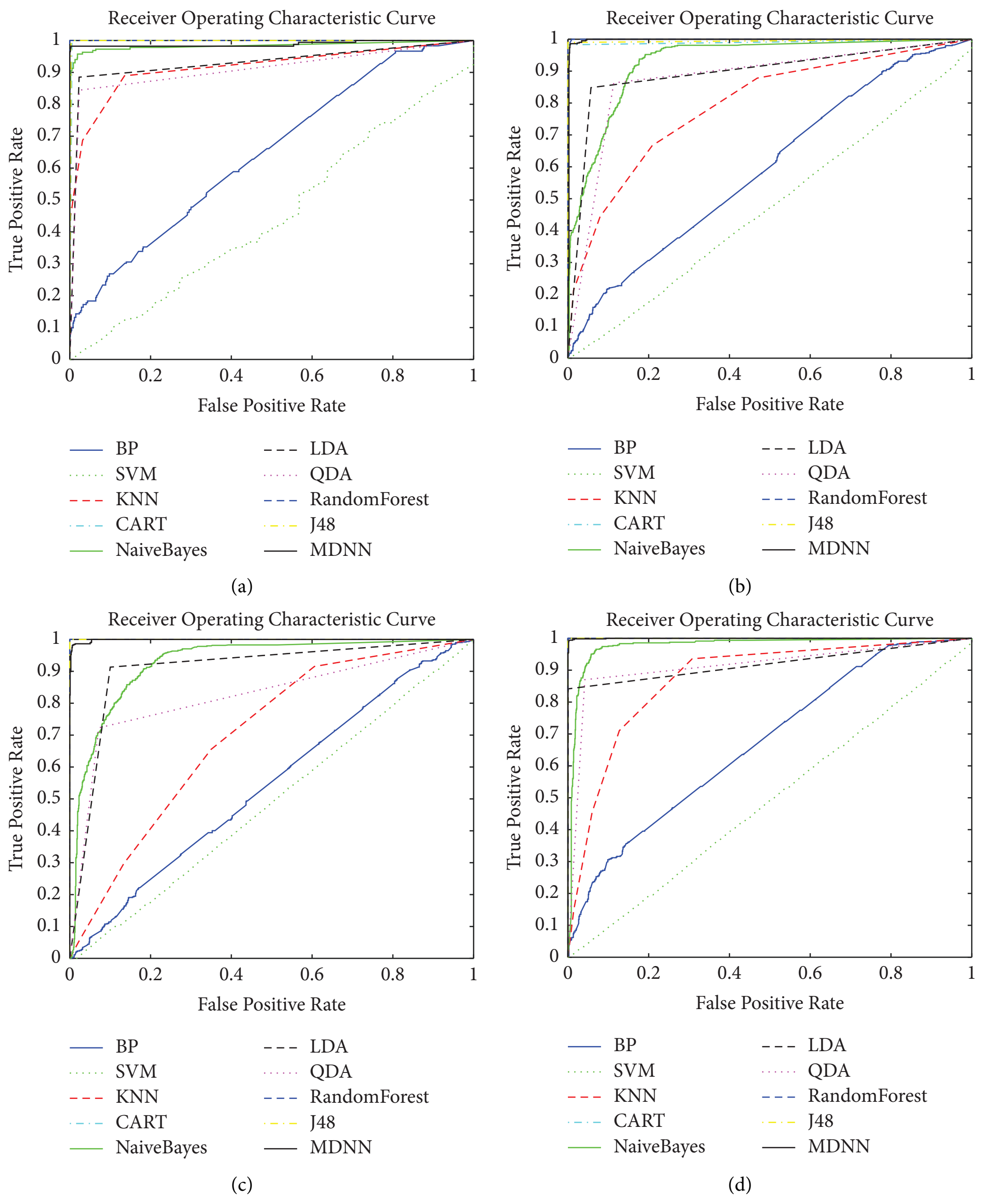

FIGURE 6: (a) The comparison of receiver operating characteristic curves for the platinum class of the QWS dataset. (b) The comparison of receiver operating characteristic curves for the gold class of the QWS dataset. (c) The comparison of receiver operating characteristic curves for the silver class of the QWS dataset. (d) The comparison of receiver operating characteristic curves for the bronze class of the QWS dataset.

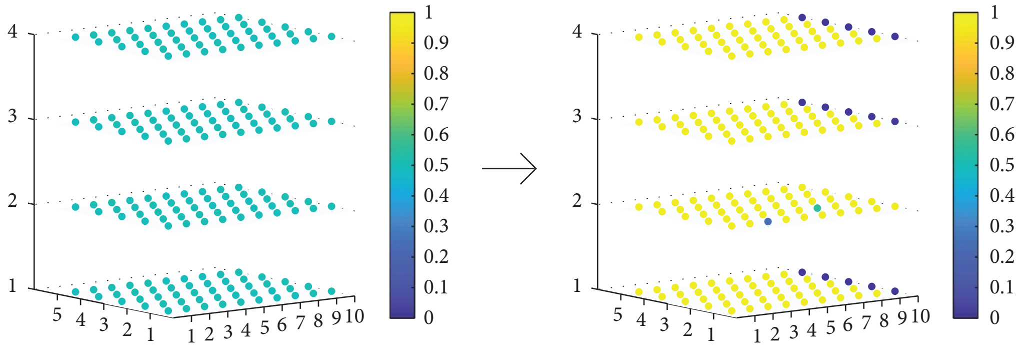

FIGURE 7: Dendritic changes in the structure of randomly selected samples before and after network training were applied to the QWS dataset. Z-axis represents the neuron, and X-axis and Y-axis are attributes and hidden layers, respectively.

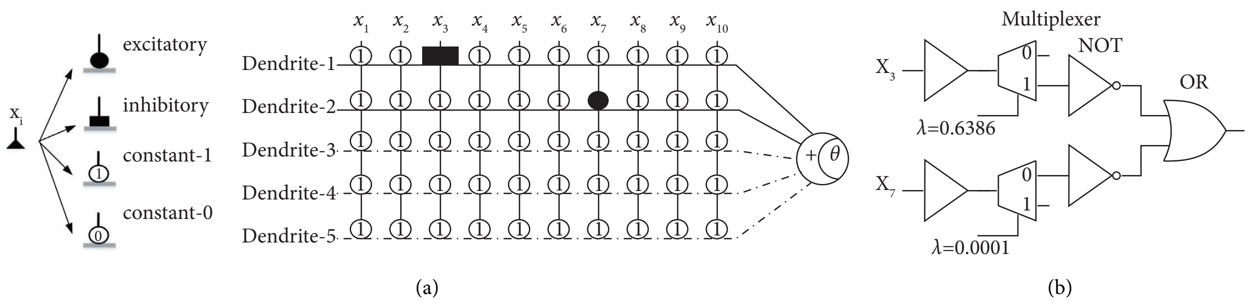

FIGURE 8: (a) The dendritic states of MDNN are fixated on the QWS dataset. Circle 0 and Circle 1 stand for constant-0 and constant-1 connection, respectively. In addition, black-filled square and circle stand for inhibitory and excitatory states, respectively. (b) Logical circle realization of MDNN on the QWS dataset, where $\lambda = q_{j,i,m}/w_{j,i,m}$.

5. Conclusion and Future Directions

This article puts forward a novel extended network of dendritic neurons, namely, the multiple dendritic neural network (MDNN). The architecture of MDNN is completely different from the previous DL models which are based on MP neuron models. By deriving its new learning algorithms, MDNN is for the first time able to resolve the multi-classification problems in comparison with previous single dendritic neuron models. Besides, we propose an approach to improve the interpretability of artificial neural networks with the theoretical support of neuroscience. Experiments are mainly carried out on a QoS-related application. In the comparison between MDNN and other classifiers, the superior performance of the proposed model is shown, and MDNN is also highly advantageous to small-scale unbalanced data. In view of this, the performance and efficiency of the proposed neural network are limited by scale. In the follow-up work, the deficiency of this experiment will be made up to improve the generalization ability [45, 46] and study the capabilities and limitations. Meanwhile, the exploration of applicable domains for MDNN will be conducted in the following aspects: 1) expanding research on more computer-related data mining to solve practical engineering problems, such as quality of service of mobile networks [47] and security bug report [48, 49]; 2) practicing in other forms of data structures, e.g., semantic [50]; 3) focusing on the unbalanced data [51] and simplifying the network structure adequately [52] with the practice.

Data Availability

The classification dataset could be downloaded freely at https://archive.ics.uci.edu/ml/index.php.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Qianyi Peng was involved in the investigation, methodology, software, visualization, validation, and writing the original draft. Shangce Gao was responsible for conceptualization, methodology, supervision, validation, and writing, review, and editing. Yirui Wang was involved in the conceptualization, software, methodology, writing, review, and editing, as well as supervision. Junyan Yi, Gang Yang, and Yuki Todo were responsible for conceptualization, methodology, and writing, review, and editing.

Acknowledgments

This research was partially supported by the JSPS KAKENHI Grant Number JP22H03643 and JP19K22891, JST SPRING Grant Number JPMJSP2145, and JST (the establishment of university fellowships towards the creation of science technology innovation) Grant Number JPMJFS2115.

References

[1] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] V. Sze, Yu-H. Chen, T.-Ju Yang, and J. S. Emer, "Efficient processing of deep neural networks: a tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[3] J. E. Dayhoff and J. M. DeLeo, "Artificial neural networks: opening the black box," *Cancer*, vol. 91, no. S8, pp. 1615–1635, 2001.

[4] Y. Yu, Z. Lei, Y. Wang, T. Zhang, C. Peng, and S. Gao, "Improving dendritic neuron model with dynamic scale-free network-based differential evolution," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, no. 1, pp. 99–110, 2022.

[5] S. Wojciech, W. Thomas, and M. . Klaus-Robert, "Explainable artificial intelligence: Understanding, Visualizing and Interpreting Deep Learning Models," 2017, https://arxiv.org/abs/1708.08296.

[6] S. Liu, Y. Xia, Z. Shi, H. Yu, Z. Li, and J. Lin, "Deep learning in sheet metal bending with a novel theory-guided deep neural network," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 3, pp. 565–581, 2021.

[7] B. Solon, H. Moritz, and N. Arvind, "Fairness in machine learning NIPS Tutorial," 2017, https://arxiv.org/abs/1810.08810.

[8] P. Lahoti, K. P. Gummadi, and G. Weikum, "ifair: learning individually fair data representations for algorithmic decision making," in *Proceedings of the 2019 IEEE 35th International Conference on Data Engineering*, pp. 1334–1345, IEEE, Macao, China, April 2019.

[9] P. Judea and D. Mackenzie, *The Book of Why: The New Science of Cause and Effect*, Basic Books, Allen Lane, London, UK, 2018.

[10] S. Bonner and F. Vasile, "Causal embeddings for recommendation," in *Proceedings of the 12th ACM Conference on Recommender Systems*, pp. 104–112, ACM, New York, NY, USA, September 2018.

[11] J. D. Olden and D. A. Jackson, "Illuminating the "black box": a randomization approach for understanding variable contributions in artificial neural networks"," *Ecological Modelling*, vol. 154, no. 1-2, pp. 135–150, 2002.

[12] M. Gevrey, I. Dimopoulos, and S. Lek, "Review and comparison of methods to study the contribution of variables in artificial neural network models," *Ecological Modelling*, vol. 160, no. 3, pp. 249–264, 2003.

[13] J. D. Olden, M. K. Joy, and R. G. Death, "An accurate comparison of methods for quantifying variable importance in artificial neural networks using simulated data," *Ecological Modelling*, vol. 178, no. 3-4, 2004.

[14] S. Warren, "Neural networks and statistical models," in *Proceedings of the Nineteenth Annual SAS Users Group International Conference*, Dallas, Texas, April 1994.

[15] B. Cheng and D. M. Titterington, "Neural networks: a review from a statistical perspective," *Statistical Science*, vol. 9, no. 1, 1994.

[16] S. Staub, E. Karaman, S. Kaya, H. Karapınar, and E. Güven, "Artificial neural network and agility," *Procedia-Social and Behavioral Sciences*, vol. 195, pp. 1477–1485, 2015.

[17] T. H. Abraham, "Physio logical circuits: the intellectual origins of the McCulloch–Pitts neural networks," *Journal of the History of the Behavioral Sciences*, vol. 38, no. 1, pp. 3–25, 2002.

[18] H. Sompolinsky, "The theory of neural networks: the hebb rule and beyond," in *Heidelberg Colloquium on Glassy Dynamics*, pp. 485–527, Springer, Berlin, Germany, 1987.

[19] S. Gao, M. C. Zhou, Z. Wang et al., "Fully complex-valued dendritic neuron model," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2021, in Press.

[20] A. Gidon, T. A. Zolnik, P. Fidzinski et al., "Dendritic action potentials and computation in human layer 2/3 cortical neurons," *Science*, vol. 367, pp. 83–87, 2020.

[21] T. Jiang, S. Gao, D. Wang, J. Ji, Y. Todo, and Z. Tang, "A neuron model with synaptic nonlinearities in a dendritic tree for liver disorders," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 12, no. 1, pp. 105–115, 2017.

[22] X. Qian, C. Tang, Y. Todo, Q. Lin, and J. Ji, "Evolutionary dendritic neural model for classification problems," *Complexity*, vol. 2020, Article ID 6296209, 13 pages, 2020.

[23] Z. Wang, S. Gao, J. Wang, H. Yang, and Y. Todo, "A dendritic neuron model with adaptive synapses trained by differential evolution algorithm," *Computational Intelligence and Neuroscience*, vol. 2020, Article ID 2710561, 19 pages, 2020.

[24] X. Luo, X. Wen, M. C. Zhou, A. Abusorrah, and L. Huang, "Decision-tree-initialized dendritic neuron model for fast and accurate data classification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 9, pp. 4173–4183, 2022.

[25] J. Ji, J. Zhao, Q. Lin, and K. C. Tan, "Competitive decomposition-based multiobjective architecture search for the dendritic neural model," *IEEE Transactions on Cybernetics*, pp. 1–14, 2022, in Press.

[26] S. Song, X. Chen, S. Song, and Y. Todo, "A neuron model with dendrite morphology for classification," *Electronics*, vol. 10, no. 9, p. 1062, 2021.

[27] S. Gao, M. Zhou, Y. Wang, J. Cheng, H. Yachi, and J. Wang, "Dendritic neuron model with effective learning algorithms for classification, approximation, and prediction," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 2, pp. 601–614, 2019.

[28] Y. Todo, Z. Tang, H. Todo, J. Ji, and K. Yamashita, "Neurons with multiplicative interactions of nonlinear synapses," *International Journal of Neural Systems*, vol. 29, no. 8, Article ID 1950012, 2019.

[29] A. Campbell, G. Coulson, and D. Hutchison, "A quality of service architecture," *ACM SIGCOMM - Computer Communication Review*, vol. 24, no. 2, pp. 6–27, 1994.

[30] A. Kaur, "An overview of quality of service computer network," *Indian Journal of Computer Science and Engineering (IJCSE)*, vol. 2, no. 3, pp. 470–475, 2011.

[31] J. W. Roberts, "Engineering for quality of service," *Self-Similar Network Traffic and Performance Evaluation*, pp. 401–420, 2000.

[32] Z. Wang and J. Crowcroft, "Quality-of-service routing for supporting multimedia applications," *IEEE Journal on Selected Areas in Communications*, vol. 14, no. 7, pp. 1228–1234, 1996.

[33] A. Tsalgatidou and T. Pilioura, "An overview of standards and related technology in web services," *Distributed and Parallel Databases*, vol. 12, no. 2, pp. 135–162, 2002.

[34] A. F. M. Huang, Ci-W. Lan, and S. J. Yang, "An optimal qos-based web service selection scheme," *Information Sciences*, vol. 179, no. 19, pp. 3309–3322, 2009.

[35] E. Al-Masri and Q. H. Mahmoud, "Discovering the best web service," in *Proceedings of the 16th International Conference on World Wide Web*, pp. 1257-1258, ACM, New York, NY, USA, May, 2007.

[36] P. K. Diederik and B. A. Jimmy, "Adam: a method for stochastic optimization," 2014, https://arxiv.org/abs/1412.6980.

[37] A. Graves, M. Abdel-rahman, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6645–6649, IEEE, Vancouver, Canada, May 2013.

[38] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proceedings of the International Conference on Machine Learning*, pp. 1139–1147, PMLR, New York, NY, USA, June 2013.

[39] I. Sutskever, *Training Recurrent Neural Networks*, University of Toronto, Toronto, Canada, 2013.

[40] G. Hinton, N. Srivastava, and S. Kevin, "Rmsprop: divide the gradient by a running average of its recent magnitude," *COURSERA: Neural Networks for Machine Learning*, vol. 13, 2012.

[41] E. Al-Masri and Q. H. Mahmoud, "Investigating web services on the world wide web," in *Proceedings of the 17th International Conference on World Wide Web*, pp. 795–804, ACM, New York, NY, USA, April 2008.

[42] E. Al-Masri and Q. H. Mahmoud, "Qos-based discovery and ranking of web services," in *Proceedings of the 2007 16th International Conference on Computer Communications and Networks*, pp. 529–534, IEEE, Honolulu, Hawaii, August 2007.

[43] S. Narkhede, "Understanding auc-roc curve," *Data Science*, vol. 26, pp. 220–227, 2018.

[44] X. Wu, V. Kumar, Q. Ross et al., "Top 10 algorithms in data mining," *Knowledge and Information Systems*, vol. 14, no. 1, pp. 1–37, 2008.

[45] K. M. R. Alam, N. Siddique, and H. Adeli, "A dynamic ensemble learning algorithm for neural networks," *Neural*

Computing & Applications, vol. 32, no. 12, pp. 8675–8690, 2020.

[46] O. Sagi and L. Rokach, "Ensemble learning: a survey," *WIREs Data Mining and Knowledge Discovery*, vol. 8, no. 4, p. e1249, 2018.

[47] G. Luo, Q. Yuan, J. Li, S. Wang, and F. Yang, "Artificial intelligence powered mobile networks: from cognition to decision," *IEEE Network*, vol. 36, no. 3, pp. 136–144, 2022.

[48] X. Wu, W. Zheng, X. Xia, and D. Lo, "Data quality matters: a case study on data label correctness for security bug report prediction," *IEEE Transactions on Software Engineering*, vol. 48, no. 7, pp. 2541–2556, 2022.

[49] W. Zheng, Y. Xun, X. Wu, Z. Deng, X. Chen, and Y. Sui, "A comparative study of class rebalancing methods for security bug report classification," *IEEE Transactions on Reliability*, vol. 70, no. 4, pp. 1658–1670, 2021.

[50] W. Zheng, X. Liu, and L. Yin, "Sentence representation method based on multi-layer semantic network," *Applied Sciences*, vol. 11, no. 3, p. 1316, 2021.

[51] W. Zheng, X. Tian, B. Yang et al., "A few shot classification methods based on multiscale relational networks," *Applied Sciences*, vol. 12, no. 8, p. 4059, 2022.

[52] L. Zhao and L. Wang, "A new lightweight network based on mobilenetv3," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 16, no. 1, 2022.