IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 8, 2021, accepted January 18, 2021, date of publication January 21, 2021, date of current version January 29, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3053280

Spike Neural Network Learning Algorithm Based on an Evolutionary Membrane Algorithm

CHUANG LIU, WANGHUI SHEN, LE ZHANG, YINGKUI DU, AND ZHONGHU YUAN
School of Information Engineering, Shenyang University, Shenyang 110044, China

Corresponding author: Chuang Liu (chuang.liu@syu.edu.cn)

This work was supported in part by the Scientific Research Funding Project of the Education Department of Liaoning Province, China under Grant 2020JYT05, in part by the Startup Research Fund for Ph.D. of Liaoning Province, China, under Grant 20170520364, in part by the Key Research and Development Program Guidance Plan of Liaoning Province, China, under Grant 2018104013, in part by the Shenyang Double Hundred Project, China under Grant 1901402, in part by the Technological Innovation Program for Young People of Shenyang City, China, under Grant RC180338, in part by the Liaoning Provincial Department of Education Innovative Talent Support Program, China under Grant LR2016074, in part by the Liaoning Provincial Key Research and Development Program, China under Grant 2019JH2/10300014, and in part by the Liaoning Revitalization Talents Program, China, under Grant XLYC1806016 and Grant XLYC1807138.

ABSTRACT As one of the important artificial intelligence fields, brain-like computing attempts to give machines a higher intelligence level by studying and simulating the cognitive principles of the human brain. Compared with the traditional neural network, the spiking neural network (SNN) has better biogenesis and stronger computing power. In this paper, an SNN learning model based on an evolutionary membrane algorithm is proposed to solve the problem of supervised classification. The proposed algorithm uses the P system's object, reaction rules, and membrane structure to solve these problems. Specifically, the proposed algorithm can automatically adjust the learning parameters of the network by adjusting the synaptic weight in the learning stage of the spiking neural model according to different application data, providing a better solution model for balance exploration and exploitation. In the simulation experiment, effectiveness verification research is carried out. The simulation results show that compared with other experimental algorithms, the proposed algorithm has a competitive advantage in solving twelve supervised classification benchmark problems through learning curves and quantified classification results.

INDEX TERMS Evolutionary membrane algorithm, P systems, spiking neural network, supervised classification.

I. INTRODUCTION

Spiking neural networks (SNNs) are often referred to as third-generation artificial neural networks, which are generalized approximate and parallel distributed processing models [1]. Different from the traditional perceptron neural network, SNN uses the spike ignition sequence for information processing and uses the spiking time coding method to encode the input variable as the spiking ignition time. The neuronal simulation of SNNs is closer to reality, and the influence of time information is also considered. The SNN neurons are not activated in each propagation iteration, and they are only activated when their membrane potential reaches a certain value. When a neuron is activated, it sends signals to other neurons to increase or decrease its membrane potential.

The associate editor coordinating the review of this manuscript and approving it for publication was Orazio Gambino.

Spike neurons are the third-generation model of artificial neuron development. Because spike neurons have the dynamic firing characteristics of biological neurons and can deal with the problem of time patterns, the study of spike neurons has become an important topic in artificial neural networks [2], [3]. SNNs provide a new highly biomimetic approach for the development of artificial neural networks (ANNs) and explore new research directions.

Compared with the continuous numerical ANN network, the SNN network has the advantages of low power consumption and high performance. Both signal transmission and synaptic computation in the network are event-driven, and signal encoding and transmission are similar to those in the biological brain, which is more biologically rational [4]. To apply the insights of biological neurons to machine learning and pattern recognition, SNNs need to be constructed first. Similar to biological neurons, the SNN processes and

transmits signals through input and output spikes. In addition, SNN uses different spike neuron models to simulate and interpret the information calculation process of biological neurons. ANN uses a bionic method to process information, and its different neuron types have different degrees of bionics. Most of the current neural network models are based on the second-generation perception neuron, which uses input variables as the firing frequency of neurons to process and transmit information. However, the results of brain science research show that information in biological systems is processed and transmitted by using precise spike firing sequences. Inspired by this discovery, SNN greatly improves SNN computing power by using the precise firing time of spikes as a network information processing method [5]. SNNs are the most biomimetic ANNs at present, have been widely studied by scholars worldwide and have solved many practical problems [6]. For example, SNNs have made a series of achievements in pattern recognition, robot navigation and control, optimization problems, associative memory and other aspects, which reflects their broad application prospects [2], [7], [8].

At present, many researchers have proposed a series of learning algorithms based on the in-depth study of the SNN model. When an SNN is employed to solve real-world problems, it needs to be configured first, including appropriate spiking neurons, network topology, and efficient learning algorithms. However, the configuration of an SNN will directly affect its generalization performance. Traditional learning algorithms such as SpikeProp are usually slow and may fall into local minima [9], which makes the performance of the network model not optimal, and the network model may have overfitting problems. These problems seriously limit the practical application of SNNs.

As SNNs are still a relatively young and active research field, currently proposed methods for training SNNs are limited. These learning algorithms mainly include the gradient descent algorithm and the evolutionary algorithm. The gradient descent algorithm used to train the ANN is extended to train the SNN. Most gradient descent methods are used only to adjust the weight of synaptic connections. However, the complexity of SNNs and the discontinuous dynamics of the spike neuron model make it very difficult to implement the gradient descent algorithm. The main disadvantage of the gradient descent algorithm is that it is sensitive to the initial state of the SNN and easily falls into a local minimum. Furthermore, if negative synaptic weights are allowed, convergence is not possible. [10]. To more effectively solve the above difficulties, using the evolutionary method configuration SNN has received increasing attention because these methods in the absence of prior knowledge in this field have the advantage of being intuitive [7], [8]. At present, evolutionary algorithms, such as genetic algorithms (GA) [11], particle swarm optimization (PSO) [12], differential evolution (DE) [13], and cuckoo search (CS) [14], are utilized to adaptively learn the configuring synaptic weights and delays of SNNs. Based on the self-learning ability of SNNs and the adaptive ability of evolutionary algorithms, an evolutionary spike neural network is proposed. An evolved SNN can automatically adjust its weights according to the sample data from the actual application. It is worth noting that each solution representing the weights of the networks is obtained in a multidimensional solution space by an evolutionary algorithm. Pavlidis *et al.* proposed the parallel differential evolution algorithm for training supervised feedforward SNNs [15]. However, the algorithm is validated using the exclusive-OR, diabetes, and Iris only, which does not reflect its advantages. Jin *et al.* proposed the Pareto-based multiobjective genetic algorithm and used latent coding to optimize the classification performance and connectivity of the spike neural network [16]. However, the performance of the multiobjective GA was not compared against other training methods. Vazquez *et al.* proposed the cuckoo search algorithm to train a spiking neuron to be applied in a pattern classification task [17]. The authors argued that the results obtained with the spiking neuron model trained with the cuckoo search algorithm were good. Vazquez *et al.* proposed a learning strategy based on artificial bee colonies to train a spiking neuron aiming to solve pattern recognition problems [18]. Saleh *et al.* proposed a hybrid harmony search algorithm with evolving SNNs for classification problems [19]. The findings demonstrate that the proposed algorithm can achieve better performance results in accuracy than the other experimental algorithms. Abusnaina *et al.* proposed the enhanced-mussels wandering optimization algorithm for pattern recognition problems [10]. The obtained results indicate that the proposed method is a competitive alternative in terms of classification accuracy and training time. Hussain *et al.* proposed an elitist floating-point genetic algorithm with a hybrid crossover algorithm to train multilayer feedforward SNNs in a supervised manner [20]. The experimental results show that the proposed algorithm has efficient computing power and biological plausibility. However, it only allows a single spike to be emitted, so it cannot be applied to an SNN that allows multiple spike times in each neuron. These experiences are very useful for improving the generalization performance of SNNs. Although these models have been successfully applied to some pattern recognition problems, there are still some major defects in the training and design stages [2], [8].

Although some preliminary research results have been obtained in the development of evolutionary SNNs, their development is still in its infancy, and there are still many problems to be solved. At present, the main problem of SNNs is the low training efficiency of the learning algorithm. To solve the above problems, this paper mainly explores a learning algorithm with high learning efficiency and high biogenesis, in line with the rules of biological cognition, a wide application range, and the ability to control the output mode of neurons. Therefore, based on previous work [13], [14], [18], [19], [21], this paper aims to study the learning algorithm based on the evolutionary membrane algorithm to further improve the generalization performance of SNNs. The evolutionary membrane algorithm is a heuristic algorithm

influenced by the internal function and structure of the cell and is an evolutionary algorithm (EA) [22], [23]. The main advantage of the evolutionary membrane algorithm is that it can adapt to the chaining environment. The evolutionary membrane algorithm is used to generate high-quality solutions to optimize and search for the solved problems by introducing a chemical reaction optimization mechanism and relying on the heuristic operations inspired by biological cells to improve the global search capability. The evolutionary membrane algorithm has been applied to many classification models, such as artificial neural networks [24]. However, the evolutionary membrane algorithm has not been applied to optimize the presynaptic neurons of the SNN. In other words, the learning algorithm is employed to learn the parameters of the SNN adaptively. Based on our previous work [22]–[24], a learning algorithm based on the evolutionary membrane algorithm is proposed to find a series of optimal solutions of SNNs. Subsequently, the learning algorithm is applied to solve 12 supervised classification problems from the University of California at Irvine (UCI) repository [25]. To verify the effectiveness of the proposed algorithm, it was compared with DE [13], CS [14], PSO [12], CRO [21], harmony search (HS) [26], and artificial bee colony (ABC) [27]. Experimental results show that the proposed algorithm is superior to similar learning algorithms in generalization performance.

The main contributions of this paper can be summarized as follows.

- For the first time, the evolutionary membrane algorithm is used as a supervised classification learning algorithm for SNNs, and its three elements are realized according to the characteristics of SNNs.
- The reaction rule provides a good SNN model through the balance of exploration and exploitation based on the search space so that the SNN output can avoid overfitting.
- In the experiment, we show that the results of the proposed algorithm are in great competition with those of the experimental algorithms. Simulation results verify the effectiveness of the proposed algorithm as an SNN learning algorithm for supervised classification problems.

The rest of this paper is summarized below. Section II discusses the concept of the Izhikevich neuron model and P system. Section III describes the proposed algorithm in detail, especially how to design objects, reaction rules and membrane structures to optimize SNNs. In Section V, the proposed algorithm is compared with the six state-of-the-art evolutionary algorithms, and the simulation results of the twelve classification problems from UCI are evaluated and discussed. Finally, Section VI summarizes the conclusions of this paper.

II. THEORETICAL BACKGROUND

A. IZHIKEVICH NEURON MODEL

The nervous system is defined by the presence of a particular type of cellular neuron. Neurons differ from other cells in many ways, but their most basic feature is that they transmit messages through synapses. To communicate with other cells, a synapse is a neuron-neuron connection that contains molecular mechanisms for rapidly transmitting electrical or chemical signals. The synaptic weight parameter determines the strength of the connection between two neurons.

Next, we discuss a more specific neuron model, the Izhikevich model [28]. The model combines the advantages of the Hodgkin-Huxley (HH) and leaky integrated-and-fire (LIF) models. The HH model was proposed by Alan Hodgkin and Andrew Huxley in 1952. Based on summarizing the data of squid nerve stimulation potential, it became the prototype of nerve cells with many different physiological structures. Although the HH model can be better explained from a biological point of view, the reality is that it is difficult to accurately measure the various parameter values of nerve cells (or the measurement cost is very expensive), which often prevents us from using this model. Therefore, from the point of view of dynamic systems, researchers have designed more concise neural models that still retain biological characteristics. The leaky integrate-and-fire model is one of them. The biological accuracy and computational complexity of the Izhikevich model are close to those of the HH model and LIF model. The HH model has the characteristics of high calculation accuracy, but the number of calculations is large. The LIF model has the characteristic of few calculations, but it sacrifices computational precision. The bifurcation method enables us to simplify many biophysically precise HH neuron models into ordinary differential equations of two-dimensional systems. Based on this, the mathematical expression of the Izhikevich model is shown in Eq.1.

$$\begin{aligned} \frac{dv}{dt} &= 0.04v^2 + 5v + 140 - u + I \\ \frac{du}{dt} &= a(bv - u) \end{aligned} \tag{1}$$

If the membrane potential is higher than 30 mV, the reset function is activated after spike emission in Eq.2.

$$if v \geq 30mV, then \begin{cases} v \leftarrow c \\ u \leftarrow u + d \end{cases} \tag{2}$$

where v and u are dimensionless variables. a, b, c, d are dimensionless parameters, and t is time. The variable v represents the membrane potential of the neuron, and u represents the membrane recovery variable, which can explain the activation of the K^+ ion current and the inactivation of the Na^+ ion current and provide negative feedback for v.

B. P SYSTEMS

With the development of P system research, many computational models are abstractly drawn from cells, tissues and neural systems. These computing models are mainly composed of cell-like, tissue-like, and neural-like models [29]–[32]. The membrane serves as the cell boundary, and the substances contained in the cell are represented as the membrane computing objects. The reaction between objects (substances)

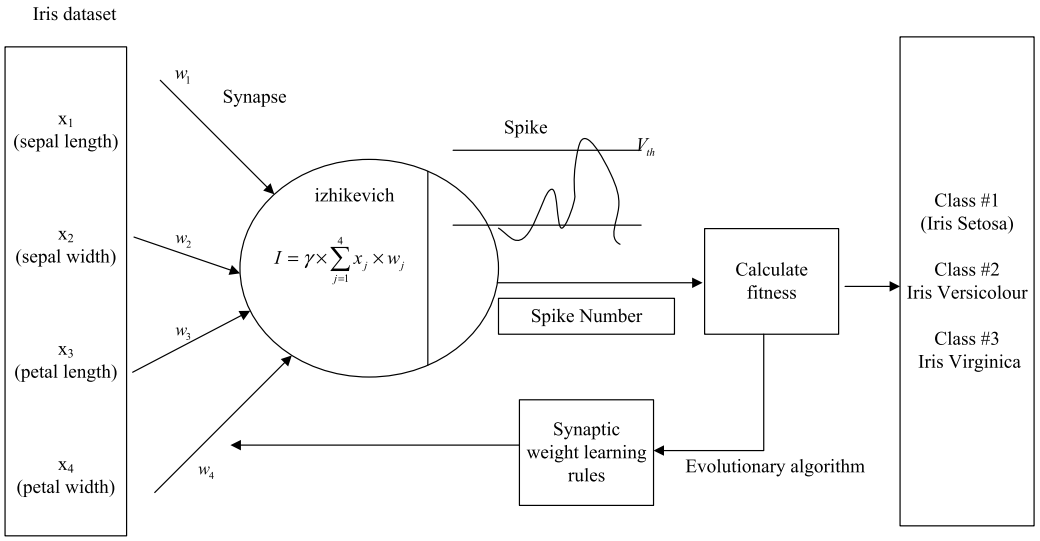

FIGURE 1. An example of the learning process of the spike neural network supervised learning algorithm.

and the flow of objects (substances) between membranes is a calculation process of the membrane algorithm, in which objects (substances) and their quantities change within the membrane region. Different changes correspond to different rules of evolution. In the P system, there are usually multiple objects and multiple reaction rules. The principles of rule execution are uncertainty and maximum parallelism [33].

If a P system is given, its membrane structure is determined. In addition, given the membrane structure, the initial model of the P system is given, and each membrane contains all corresponding objects and evolutionary rules. Starting with the initial pattern, the individual membranes of the P system use the rules in an uncertain maximum parallel manner. Each time the P system runs, it changes the current pattern accordingly and enters a new pattern. After many computations, the P system forms a series of patterns. When there are no executable rules, the system shuts down. The whole process is called the calculation of the P system, and the object sent to the environment or the specified membrane is the result of the calculation.

III. PROPOSED FRAMEWORK

In this section, we propose a new evolutionary algorithm under the framework of the P system, namely, the supervised learning algorithm of the spike neural network. The proposed algorithm consists of objects, reaction rules and membrane structures. First, the object is initialized, and the initialized objects are divided into multiple object sets. Second, each membrane updates the object by invoking a different reaction rule while preserving all objects discovered by the membrane. To promote the interaction between the membranes, the best objects in the membrane are exchanged periodically. Then, the globally optimal object of each membrane is updated according to the Lévy flight, normal distribution and uniform distribution. Finally, all objects are output to the skin membrane region. The specific pseudocode of the proposed algorithm is shown in Algorithm 1.

Algorithm 1 The Pseudocode of the Proposed Algorithm

Require: Let N be the object size, Max_Gen be the number of iterations, G be the number of elementary membranes, and A be the evolved object set.

Ensure: Final Objects

```
Objects ← Initialization(N)
Evaluate(Objects)
A ← Objects
for i = 1; i < Max_Gen; i++ do
    for j = 1; j < G; j++ do
        Mem(j) ← Evolving(Mem(j))
        Evaluate(Mem(j))
        A ← A ⋃ Mem(j)
    end for
    Objects ← A(1 : N, :)
end for
Final Objects
```

Taking the Iris dataset as an example, Figure 1 introduces the training process of the supervised learning algorithm of the spiking neural network.

The procedure in Figure 1 can be explained as follows:

- The four features of the Iris dataset serve as inputs to the Izhikevich neuron model
- The decision variables of the evolutionary algorithm correspond to the connection weights of Izhikevich neurons
- After calculation by the Izhikevich neuron model, it was determined whether the threshold of membrane potential was reached to excite the spike
- The possible classification of the current input characteristics is calculated according to the number of excitation spikes

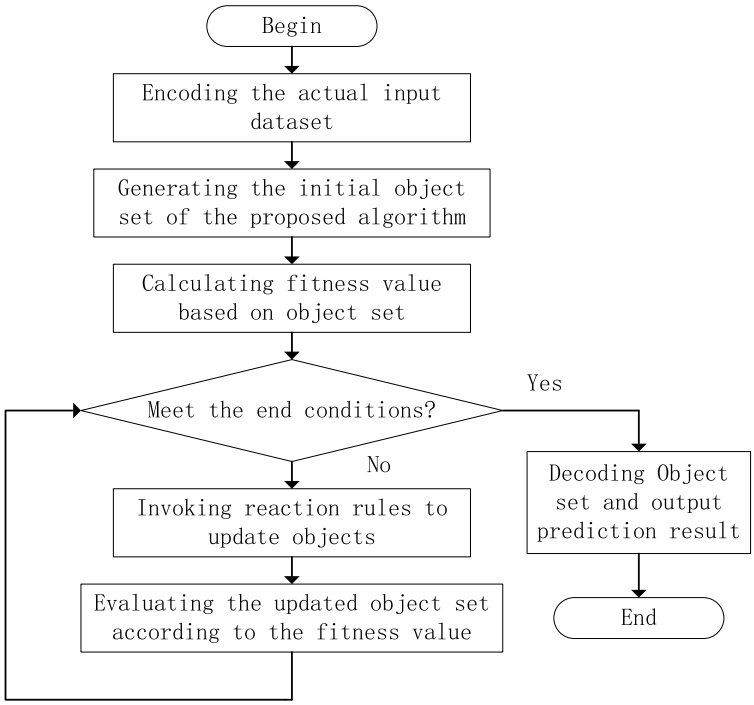

FIGURE 2. The flowchart of the proposed algorithm.

- According to the prediction result error, the evolutionary algorithm is called to generate new weights until the optimal classification results appear

The flowchart of the proposed algorithm is given in Figure 2.

The pseudocode of the proposed algorithm is described in Algorithm 1. Its specific characteristics are defined and explained in the following sections.

A. OBJECTS

In the proposed algorithm, the object is the decision variable, which is used to represent the synaptic weights in the spiking neural network. To restrict the optimization range of the object, it is necessary to give the upper bound and the lower bound of the synaptic weights in the spiking neural network. In the proposed algorithm, the upper bound of the object is set as 20, and the lower bound is set as -20. After initializing the parameters of the P system, the skin membrane is established. An object in the skin membrane can be initialized according to Eq. (3).

$$\begin{aligned} w_{i,j} &= w_j^l + (w_j^u - w_j^l) \times r \\ 1 &\le i \le N \\ 1 &\le j \le D \end{aligned} \tag{3}$$

where N represents the total number of objects. D is the dimension of a decision variable. $w_{i,j}$ is the value of the j-th dimension in the i-th object. w_j^l is the lower value in the j-th dimension of weight, and w_j^u represents the upper value in the j-th dimension of weight. r denotes a random number on the interval (0, 1).

B. MEMBRANE STRUCTURE

The proposed algorithm takes the cell-like structure as the initial mode of the P system. The structure of the proposed algorithm is a two-layer structure composed of the skin membrane and several membranes. The skin membrane is the outermost membrane of the membrane structure, which contains objects within its region and evolutionary rules. In the P system, a single membrane of the P system will make use of the evolutionary rule within the membrane in an indeterminate, maximum parallel manner when the initial mode of the P system is determined. This helps speed up the optimization of the proposed algorithm. To simplify the realization of the membrane structure of the P system, the proposed algorithm only uses a structure containing four elementary membranes in the skin membrane.

C. REACTION RULES

In the proposed algorithm, a mechanism of the chemical reaction optimization (CRO) algorithm calling rules is introduced to evolve the objects in the membrane. These reaction rules of CRO are reimplemented as the reaction rules of the proposed algorithm to simulate the process of evolving objects of molecular interaction. CRO consists of four basic reaction rules, namely, onWall, decomposition, interaction and synthesis, with self-organization, self-assembly and other functions. If objects in the membrane collide, one of these

Algorithm 2 Pseudocode for the Calling Logic of These Rules

Require: N: The number of objects (molecules); *Objects*: The set of objectives.
Ensure: O:The evolved objectives.

```
while sizeof(Objects) ≠ 0 do
    if random > Molecule_Collision||i == N then
        Randomly select an object(o_i) from Objects and delete it from Objects
        if o_i.numofhits > alpha then
            O = O ⋃ decomposition( o_i);
        else
            O = O ⋃ onWall(o_i);
        end if
    else
        Randomly select two objects(o_i, o_j) from Objects and delete them from Objects
        if o_i.KE <= beta&&o_j.KE <= beta then
            O = O ⋃ synthesis( o_i, o_j);
        else
            O = O ⋃ interaction( o_i, o_j);
        end if
    end if
end while
return O
```

four rules is called. In the proposed algorithm, there are only two cases of single object evolution and two object evolution in a membrane. The largest number of colliding molecules in the membrane is 2. We provide pseudocode for these rules in Algorithm 2. Refer to the literature [21] for the specific implementation form of these four rules. For an object, Eq.4 gives the evolution rule for a single object. The object in the cell has the characteristics of random walk or Brownian motion. The proposed algorithm introduces the Lévy flight to change the information of different dimensions of the object.

$$w' = w + \alpha \oplus Lévy(\lambda) \tag{4}$$

The proposed algorithm use $\alpha = 1$ and $\lambda = 1.2$.

For both objects, Eq.5 gives the evolution rules for both objects. We use a normal distribution or random variate to express the randomness of the object in the membrane region.

$$\begin{aligned} w_1' &= \begin{cases} w_2 + (w_2 - w_o) * normrnd * 0.12, & r < 0.1 \\ w_1 + (w_o - w_2) * r, & otherwise \end{cases} \\ w_2' &= \begin{cases} w_1 + (w_1 - w_o) * normrnd * 0.12, & r < 0.1 \\ w_2 + (w_o - w_1) * r, & otherwise \end{cases} \end{aligned} \tag{5}$$

IV. FITNESS FUNCTION

To evaluate the object of the proposed algorithm, it is necessary to design the fitness function of the learning algorithm for the SNN model. The fitness function is based on the synaptic weights of the Izhikevich neuron model (the object of the proposed algorithm). The synaptic weight of the SNN was adjusted by implementing reaction rules, and the firing rate of neurons was generated by the Izhikevich neuron model. Better synaptic weights can be generated by this fitness function in Eq.6.

$$\begin{aligned} \bar{r}_j &= \sum_{i=1}^{B} \frac{n_{i,j}}{B}, \\ f &= \frac{\sum_{j=1}^{C} argmin|n - \bar{r}_j|}{B} \\ 1 &\le i \le B, 1 \le j \le C \end{aligned} \tag{6}$$

where $\bar{r}$ is the average fire rate of the Izhikevich neuron model. $n_{i,j}$ is the spike number of the i-th sample and j-th class label. B is the total number of classification data. C is the total number of classification labels.

V. EXPERIMENTAL STUDIES

To verify the training effect of the algorithm mentioned in this paper, benchmark datasets are selected from the UCI Machine Learning Repository to test the learning effect of the proposed algorithm. With regard to this, we selected twelve supervised classification benchmarks to analyze the performance of the proposed algorithm and provide insights. First, we describe the benchmark function and evaluation indicator used in the experiment. Second, we give a brief comparison of the latest experimental algorithms and provide the experimental setup used in this study. Finally, the comparison results of experimental algorithms under different benchmark functions are discussed.

A. TEST PROBLEMS AND EVALUATION INDICATORS

1) TEST PROBLEMS

The experiment involves 12 benchmark datasets from the UCI Machine Learning Repository [25]. The datasets used in the experiment included Iris, Wine, Diabetes, Heartstatlog, Ionosphere, Sonar, Aggregation, Vowel, Bupa, Cancer, Thyroid,

TABLE 1. The details of the twelve benchmark datasets.

Dataset	Total Samples	Train Samples	Test Samples	Classes	Features
Iris Plant	150	105	45	3	4
Wine	178	105	73	3	13
Diabetes	768	537	231	2	8
Heartstatlog	270	189	81	2	13
Ionosphere	351	245	106	2	34
Sonar	208	144	64	2	60
Aggregation	788	548	240	3	2
Vowel	871	606	265	6	3
Bupa	375	271	104	2	6
Cancer	683	477	206	2	9
Thyroid	215	150	65	3	5
WDBC	569	397	172	2	30

and WDBC. Table1 gives details of the twelve benchmark datasets.

Each dataset is divided into two subsets, one of which is allocated as the training set and the other as the test set. The training dataset is allocated for SNN to find the best parameters, and the test dataset is used to prove the performance of the best solution provided by the proposed algorithm and the test dataset is used to demonstrate the algorithm's performance in providing the best solution.

We designed a cross-validation function. Its function is to randomly select train data and test data from the sample in proportion. The dataset is divided into a training set and a test set, and the category ratio of the training set and test set is required to be the same as that of the original dataset. The category ratio of the training sets is generally greater than or equal to 0.7. The input parameters of the function are the dataset matrix, corresponding class label, number of classes, and proportion of the training dataset. The training dataset and corresponding class label are returned, and the test set and corresponding class tags are also determined.

2) EVALUATION INDICATORS

The classification performance indicator (CPI) can be used to judge the performance of related algorithms. Its expression is shown in Eq.7.

$$CPI = \frac{n}{N} \tag{7}$$

where n is the number of correct classifications by evolving SNN. N is the total number of actual classifications.

3) EXPERIMENTAL CONDITIONS

We chose six experimental evolutionary algorithms, DE [13], CS [14], PSO [12], HS [19], ABC [18], and CRO [21], to compare with the proposed algorithm to evaluate its performance.

CPI is used to compare the solving performance of all experimental algorithms. It is worth noting that a higher experimental algorithm value in CPI means better performance on different benchmark datasets from the UCI Machine Learning Repository.

All experiments run on a Windows 10 Professional Edition with a 2.5 GHz Intel Pentium quad-core i7-4710MQ and 12GB of RAM. All experimental algorithms are implemented in MATLAB 2018b.

TABLE 2. The parameters of Izhikevich neuron model.

Neuron Model	Parameter settings	
	Parameter	Value
Izhikevich	C	100
	v_r	60
	v_t	-40
	k	0.7
	a	0.03
	b	-2
	c	-50
	d	100
	v_{peak}	35

Among the neuron models, the Izhikevich model has the advantages of high-precision and simple calculation compared with other neuron models. In this experiment, the Izhikevich model was selected. The parameters of the Izhikevich neuron model are set in Table2.

Because the results of evolutionary algorithms are characterized by a random search process, 20 repeats are performed on each of twelve benchmark datasets, and statistical results are attained to evaluate the statistical performance of these algorithms. The population size of all algorithms is 40, and the maximum number of iterations is set to 20. The boundary value of each solution is set from -20 to 20.

B. RESULTS OF SUPERVISED CLASSIFICATION PROBLEMS

The performance of the proposed algorithm is evaluated by comparing the CPI value with that of the experimental algorithm. The statistical results of CPI are used to judge the performance of all experimental algorithms.

1) COMPARING THE RESULTS OF EVOLUTIONARY LEARNING ALGORITHMS

Figure1 shows the learning process of all experimental algorithms on 12 supervised classification datasets. As seen from Figure 3, the change in the learning accuracy value during the iteration process can be seen. It can be seen that with the increase in iteration times, all the experimental algorithms finally converge. We use the convergence of different colors to represent the learning curve results of the experimental algorithms. The red line represents the learning curve of the proposed algorithm, the blue line represents the DE algo-

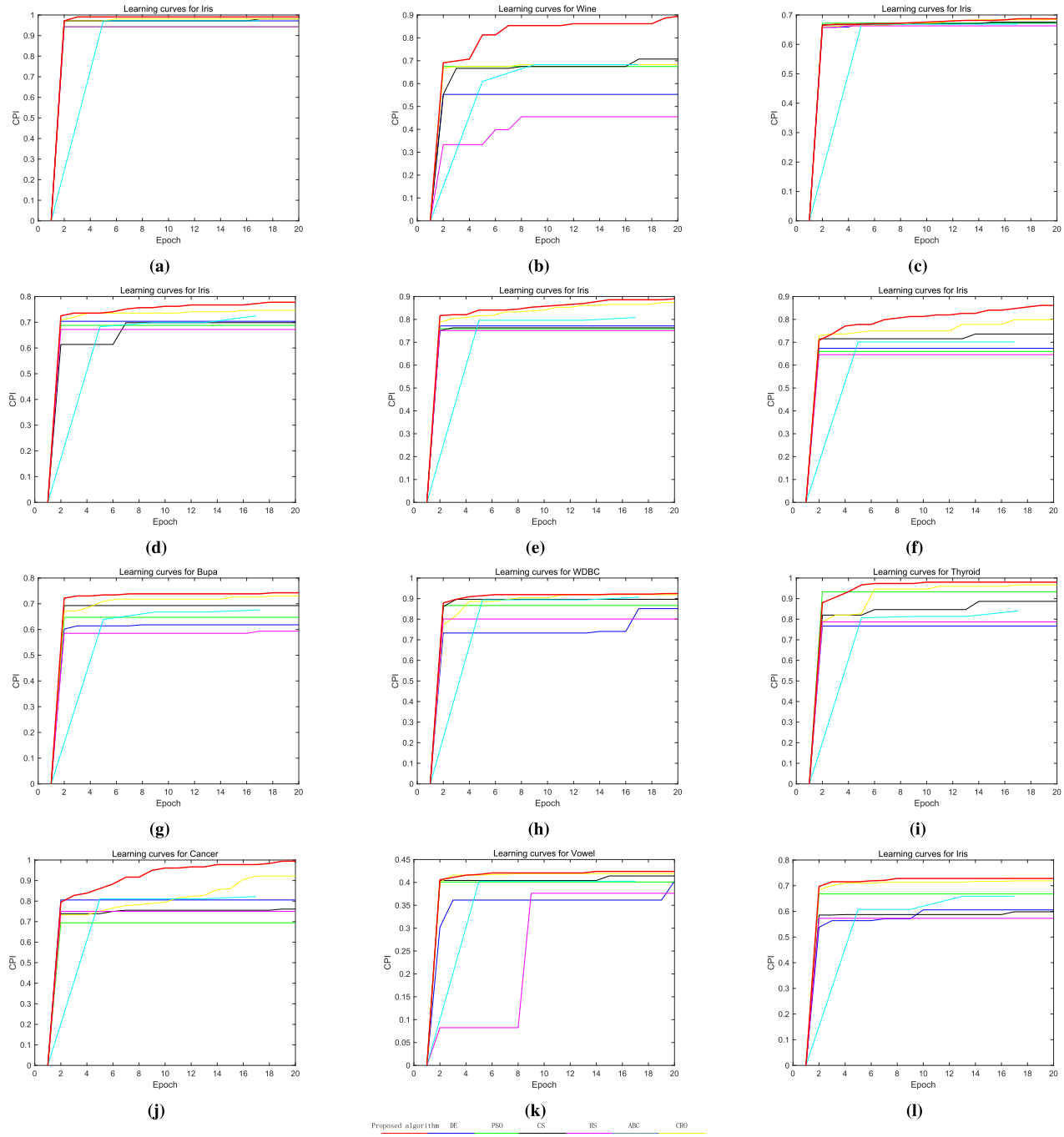

FIGURE 3. The learning curve of the proposed algorithm on 12 datasets.

rithm, the green line represents the PSO algorithm, the black line represents the CS algorithm, the purple line represents the HS algorithm, and the blue and green lines represent the ABC algorithm. It can be seen from the learning curve that the proposed algorithm is superior to these experimental algorithms in 12 supervised classification datasets, which is related to the membrane structure and reaction rules. These mechanisms can balance exploration and exploitation well, help jump out of local optimality and find approximate global optimality.

To further analyze the classification results of the proposed algorithm in the supervised classification problems, Figure 4 gives the confusion matrix of the proposed algorithm on the 12 supervised classification datasets. In Figure 4a, the proposed algorithm misclassifies a sample of category 2 into category 1 in the Iris dataset. In Figure4b, the algorithm we proposed misclassified 5 samples of category 2 and 9 samples of category 3 into category 1 in the Wine dataset. In Figure4c, the proposed algorithm misclassified 16 samples of category 2 into category 1 in the Diabetes dataset. In

FIGURE 4. The confusion matrix of the proposed algorithm on 12 datasets.

TABLE 3. Statistical results of experimental algorithms for training datasets on CPI.

FUN	DE [13]		PSO [12]		CS [14]		HS [19]		ABC [18]		CRO [21]		Proposed Algorithm	
	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD
Iris	93.38	0.04451	90.9	0.06522	94.62	0.01584	93.71	0.02986	97.33	0.005863	93.52	0.02405	**98.86**	0.004259
Wine	68.13	0.02469	47.4	0.1328	70.93	0.02189	61.87	0.07257	67.48	0.02159	69.07	0.03331	**80.81**	0.08539
Diabetes	67.33	0.004163	65.85	0.00204	67.95	0.006532	66.3	0.003796	67.22	0.00462	67.7	0.006741	**69.94**	0.005976
Heartstatlog	71.27	0.01417	64.92	0.02626	74.52	0.02588	66.3	0.03147	71.8	0.01611	73.44	0.03154	**83.17**	0.04664
Ionosphere	81.69	0.01365	81.24	0.02725	84.61	0.01475	77.06	0.01953	80.88	0.0154	81.61	0.01641	**90.29**	0.01741
Sonar	72.6	0.01569	70.42	0.02106	77.47	0.02292	67.85	0.02956	72.05	0.01669	72.85	0.02195	**87.78**	0.02888
Aggregation	59.34	0.03508	39.78	0.1978	65.51	0.03549	71.09	0.02914	57.77	0.06786	67.67	0.0298	**71.38**	0.03611
Vowel	40.19	0.01253	9.942	0.0544	40.8	0.006554	41.69	0.007366	37.27	0.03485	40.99	0.009538	**43.45**	0.02243
Bupa	65.25	0.02896	48.63	0.07538	67.53	0.02158	71.91	0.01622	63.76	0.02454	69.0	0.01703	**74.09**	0.004755
Cancer	96.02	0.01032	67.78	0.2057	96.69	0.005551	97.61	0.005286	95.85	0.008484	97.13	0.002643	**98.01**	0.001983
Thyroid	83.1	0.03996	70.0	0.002278	87.67	0.03914	93.03	0.03333	81.67	0.04781	89.33	0.03474	**97.1**	0.01487
WDBC	85.44	0.05158	60.84	0.07102	90.49	0.01831	92.02	0.006543	84.03	0.07701	91.03	0.007982	**93.17**	0.004616

TABLE 4. Statistical results of experimental algorithms for testing datasets on CPI.

FUN	DE [13]		PSO [12]		CS [14]		HS [19]		ABC [18]		CRO [21]		Proposed Algorithm	
	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD
Iris	95.11	0.03718	92.56	0.07	96.0	0.03351	95.89	0.02724	97.89	0.02442	94.0	0.04271	**97.78**	0.02722
Wine	64.82	0.08073	51.64	0.1583	66.82	0.06267	63.27	0.07476	65.73	0.05907	66.82	0.05592	**73.82**	0.08192
Diabetes	59.68	0.1178	53.38	0.1305	63.59	0.06863	56.54	0.1279	54.18	0.1355	64.13	0.0667	**66.84**	0.0117
Heartstatlog	67.41	0.02954	63.83	0.03131	69.07	0.0348	64.07	0.046	67.47	0.02812	66.85	0.05496	**77.53**	0.06259
Ionosphere	80.75	0.04958	79.06	0.03731	82.03	0.02711	75.61	0.049	79.39	0.05142	78.58	0.0603	**82.45**	0.03028
Sonar	67.03	0.0839	65.31	0.07665	70.08	0.05536	63.28	0.0528	64.14	0.05005	66.64	0.0675	**72.81**	0.0237
Aggregation	53.1	0.1211	40.15	0.1981	64.38	0.05345	63.02	0.0423	55.81	0.09329	**66.29**	0.04253	61.83	0.05952
Vowel	38.58	0.02824	9.962	0.05353	38.68	0.03354	38.66	0.03125	34.64	0.05263	39.04	0.02252	**40.51**	0.02428
Bupa	62.36	0.06671	48.51	0.06702	63.8	0.06479	67.79	0.04428	59.95	0.0683	66.39	0.04787	**70.38**	0.01861
Cancer	95.73	0.01236	70.02	0.187	96.14	0.01341	96.67	0.01301	95.24	0.01715	**97.21**	0.007855	96.75	0.00632
Thyroid	81.85	0.0787	69.23	0.00227	84.38	0.09071	90.46	0.1604	79.92	0.1578	85.31	0.1689	**97.0**	0.02519
WDBC	84.88	0.05275	61.45	0.06127	88.84	0.02471	90.09	0.01572	83.28	0.07592	90.17	0.02083	**90.64**	0.01206

Figure4d, the proposed algorithm misclassified 25 samples of category 2 into category 1 in the Heartstatlog dataset. In Figure4e, the proposed algorithm misclassified 9 samples of category 2 into category 1 in the ionosphere dataset. In Figure4f, the proposed algorithm misclassified 9 samples of category 2 into category 1 in the Sonar dataset. In Figure4g, the proposed algorithm misclassified 13 samples of category 2 into category 1 in the Bupa dataset. In Figure4h, the proposed algorithm misclassified 26 samples of category 2 into category 1 in the WDBC dataset. In Figure4i, the proposed algorithm misclassified 21 samples of category 2 into category 1 in the Thyroid dataset. In Figure 4j, the proposed algorithm misclassified 1 sample of category 2 into category 1 in the Cancer dataset. In Figure 4k, the proposed algorithm is consistent with the predicted value and actual value on category 1 and category 3, while 51 samples of category 2 and 1 sample of category 6 are misclassified into category 1, but the worst is that all samples of categories 4 and 5 are incorrectly classified into category 1. In Figure 4l, the proposed algorithm is consistent with the predicted value and actual value on category 1 and category 5, while 21 samples of category 2, 5 samples of category 3, 67 samples of category 4, 37 samples of category 6, and 11 samples of category 7 are misclassified into category 1.

The above results may be due to the small number of training samples for these categories. The number of training samples will directly affect the learning ability of the feature. When the training samples are too small, they cannot learn enough information to correctly identify the corresponding classification.

In addition, to verify the effectiveness of the proposed algorithm, we conducted experiments and compared the quantitative results with other experimental algorithms. Table 3 gives the comparison results obtained by all experimental algorithms of the training datasets on CPI. It can be seen that the proposed algorithm is superior to other algorithms in all supervised classification benchmark problems. We can easily see that the CS algorithm is superior to DE, PSO, and CRO. CRO also obtains good results on more train datasets. Compared with other algorithms, PSO has the worst results on all training datasets.

Above are the simulation results based on the training datasets. To further verify the effectiveness of the proposed algorithm, test datasets are used to verify the trained network. The simulation results of CPI for test datasets are shown in Table 4. Compared with other experimental algorithms, the proposed algorithm can still obtain the optimal results on most supervised classification benchmark test datasets. As seen from the simulation results in Table 4, the results of CS are ranked second among other algorithms other than Wine and Diabetes. CRO had the best results in aggregation and cancer, and it also had good results, especially in the Wine and Diabetes dataset. Clearly, CRO is superior to DE and PSO on all test datasets.

2) DISCUSSIONS

In summary, we selected 12 supervised classification datasets from different application fields to verify the effectiveness of the proposed algorithm. In the simulation experiment of the learning curve in Figure 3, the results of the proposed algorithm are compared with those of the experimental algorithms. We can easily find that the proposed algorithm has a faster convergence rate, especially for the 2-classification problem. By observing the results of the confusion matrix in Figure 4, we can see that the proposed algorithm can achieve good results in most supervised classification datasets, but in some categories with fewer samples, the prediction accuracy may be affected to some extent. The simulation results in Tables 3 and 4 show that the proposed algorithm is effective for training SNNs and can generate

better weights of SNN to avoid overfitting. Other experimental algorithms make the network model perform poorly. According to the above experimental results, we can conclude that the proposed algorithm is an effective use of the three elements of the membrane algorithm. The proposed algorithm is insensitive to the variation in various parameters, which is conducive to the wide application of various problems. However, the proposed algorithm does not perform well in the problem of classification above three class labels and the problem of a small number of samples.

VI. CONCLUSION

The research work in this paper is to improve the prediction accuracy of pulsed neural networks in supervised classification problems by combining the P system and chemical reaction optimization methods. According to the characteristics of the spike neural network problem, an effective learning algorithm is proposed to realize the three elements of the P system, which improves the accuracy and robustness of the prediction model on supervised classification problems. Furthermore, the proposed algorithm introduces three elements of the P system to find the optimal weight of the SNN. These mechanisms enhance the diversity of candidate solutions, avoid local minima, improve the global optimization ability of the algorithm, and thus avoid the SNN overfitting problem.

Experimental results show that the proposed algorithm is effective and has advantages over the simulation algorithms in the experiment. Therefore, in this paper, SNNs based on evolutionary membrane algorithm optimization are effectively applied to supervised classification, which promotes the development of SNNs in the application field. Using the evolutionary membrane algorithm to learn the parameters of SNNs can improve the theoretical research of network supervised learning algorithms. In addition to the greedy learning algorithm, the work of this paper also provides a new perspective for SNN learning on supervised classification problems.

Because SNNs are in the stage of rapid development, they have shown great computing power and application prospects. In this paper, the precision of the learning algorithm of SNNs is solved. Before SNNs can be successfully applied to various tasks, there are still many problems to be solved, including the design of learning algorithms for complex deep network structures, the study of multineuron and multispike network learning algorithms, and the improvement of the reaction rules of the learning algorithm to enhance the classification accuracy for the supervised classification of small samples.

ACKNOWLEDGMENT

The authors would like to thank the editors and reviewers for providing useful comments and suggestions to improve the quality of this article.

REFERENCES

[1] J. J. Wade, L. J. McDaid, J. A. Santos, and H. M. Sayers, "SWAT: A spiking neural network training algorithm for classification problems," *IEEE Trans. Neural Netw.*, vol. 21, no. 11, pp. 1817–1830, Nov. 2010.
[2] J. L. Lobo, J. D. Ser, A. Bifet, and N. Kasabov, "Spiking neural networks and online learning: An overview and perspectives," *Neural Netw.*, vol. 121, pp. 88–100, Jan. 2020.
[3] L. Zhang, "Building logistic spiking neuron models using analytical approach," *IEEE Access*, vol. 7, pp. 80443–80452, 2019.
[4] S. Yang, J. Wang, B. Deng, C. Liu, H. Li, C. Fietkiewicz, and K. A. Loparo, "Real-time neuromorphic system for large-scale conductance-based spiking neural networks," *IEEE Trans. Cybern.*, vol. 49, no. 7, pp. 2490–2503, Jul. 2019.
[5] N. Rathi, P. Panda, and K. Roy, "STDP-based pruning of connections and weight quantization in spiking neural networks for energy-efficient recognition," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 4, pp. 668–677, Apr. 2019.
[6] S. Guo, Z. Yu, F. Deng, X. Hu, and F. Chen, "Hierarchical Bayesian inference and learning in spiking neural networks," *IEEE Trans. Cybern.*, vol. 49, no. 1, pp. 133–145, Jan. 2019.
[7] S. Schliebs and N. Kasabov, "Evolving spiking neural network—A survey," *Evolving Syst.*, vol. 4, no. 2, pp. 87–98, 2013.
[8] X. Wang, X. Lin, and X. Dang, "Supervised learning in spiking neural networks: A review of algorithms and evaluations," *Neural Netw.*, vol. 125, pp. 258–280, May 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608020300563
[9] S. Dora, S. Sundaram, and N. Sundararajan, "An interclass margin maximization learning algorithm for evolving spiking neural network," *IEEE Trans. Cybern.*, vol. 49, no. 3, pp. 989–999, Mar. 2019.
[10] A. A. Abusnaina, R. Abdullah, and A. Kattan, "Supervised training of spiking neural network by adapting the E-MWO algorithm for pattern classification," *Neural Process. Lett.*, vol. 49, no. 2, pp. 661–682, 2019.
[11] M. Srinivas and L. M. Patnaik, "Genetic algorithms: A survey," *Computer*, vol. 27, no. 6, pp. 17–26, Jun. 1994.
[12] C. A. C. Coello, G. T. Pulido, and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 256–279, Jun. 2004.
[13] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.
[14] X.-S. Yang and S. Deb, "Cuckoo search: Recent advances and applications," *Neural Comput. Appl.*, vol. 24, no. 1, pp. 169–174, Jan. 2014.
[15] N. G. Pavlidis, O. K. Tasoulis, V. P. Plagianakos, G. Nikiforidis, and M. N. Vrahatis, "Spiking neural network training using evolutionary algorithms," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, vol. 4, Jun. 2005, pp. 2190–2194.
[16] Y. Jin, R. Wen, and B. Sendhoff, "Evolutionary multi-objective optimization of spiking neural networks," in *Proc. Int. Conf. Artif. Neural Netw.* Berlin, Germany: Springer, 2007, pp. 370–379.
[17] R. A. Vazquez, "Training spiking neural models using cuckoo search algorithm," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jun. 2011, pp. 679–686.
[18] R. A. Vazquez and B. A. Garro, "Training spiking neural models using artificial bee colony," *Comput. Intell. Neurosci.*, vol. 2015, pp. 1–14, Jul. 2015.
[19] A. Y. Saleh, S. M. Shamsuddin, H. N. A. Hamed, T. C. Siong, and M. K. Othman, "A new harmony search algorithm with evolving spiking neural network for classification problems," *J. Telecommun., Electron. Comput. Eng.*, vol. 9, nos. 3–11, pp. 23–26, 2017.
[20] I. Hussain and D. M. Thounaojam, "SpiFoG: An efficient supervised learning algorithm for the network of spiking neurons," *Sci. Rep.*, vol. 10, no. 1, pp. 1–11, Dec. 2020.
[21] A. Y. S. Lam and V. O. K. Li, "Chemical-reaction-inspired metaheuristic for optimization," *IEEE Trans. Evol. Comput.*, vol. 14, no. 3, pp. 381–399, Jun. 2010.
[22] C. Liu and Y. Du, "A membrane algorithm based on chemical reaction optimization for many-objective optimization problems," *Knowl.-Based Syst.*, vol. 165, pp. 306–320, Feb. 2019.
[23] C. Liu, Y. Du, A. Li, and J. Lei, "Evolutionary multi-objective membrane algorithm," *IEEE Access*, vol. 8, pp. 6020–6031, 2020.
[24] C. Liu, D. Chen, and F. Wan, "Multiobjective learning algorithm based on membrane systems for optimizing the parameters of extreme learning machine," *Optik*, vol. 127, no. 4, pp. 1909–1917, Feb. 2016.
[25] D. Newman. (2007). *UCI Machine Learning Repository*. [Online]. Available: http://archive.ics.uci.edu/ml/index.php

[26] M. G. H. Omran and M. Mahdavi, "Global-best harmony search," *Appl. Math. Comput.*, vol. 198, no. 2, pp. 643–656, May 2008.
[27] B. Dervis and B. Karaboga, "Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems," *Found. Fuzzy Log. Soft Comput.*, vol. 11, no. 3, pp. 789–798, 2007.
[28] J. Fil and D. Chu, "Minimal spiking neuron for solving multilabel classification tasks," *Neural Comput.*, vol. 32, no. 7, pp. 1–22, 2020.
[29] X. Zhang, L. Pan, and A. Paun, "On the universality of axon P systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 11, pp. 2816–2829, Nov. 2015.
[30] G. Singh and K. Deep, "A new membrane algorithm using the rules of particle swarm optimization incorporated within the framework of cell-like P-systems to solve Sudoku," *Appl. Soft Comput.*, vol. 45, pp. 27–39, Aug. 2016.
[31] I. Pérez-Hurtado, D. Orellana-Martín, G. Zhang, and M. J. Pérez-Jiménez, "P-lingua in two steps: Flexibility and efficiency," *J. Membrane Comput.*, vol. 1, no. 2, pp. 93–102, Jun. 2019.
[32] P. Guo, C. Quan, and H. Chen, "MEAMVC: A membrane evolutionary algorithm for solving minimum vertex cover problem," *IEEE Access*, vol. 7, pp. 60774–60784, 2019.
[33] H. Peng and J. Wang, "Coupled neural P systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 6, pp. 1672–1682, Jun. 2019.

CHUANG LIU received the Ph.D. degree in control theory and control engineering from the Dalian University of Technology, Dalian, in 2014. He has made a deep research on the evolutionary computation and membrane computing. He is currently an Associate Professor with the School of Information Engineering, Shenyang University. His current research interests include multiobjective optimization, evolutionary computation, evolutionary neural networks, the Internet of Things, and modeling and control in complex industrial systems.

WANGHUI SHEN is currently pursuing the M.S. degree in control theory and control engineering with Shenyang University, China. His current research interests include multiobjective optimization, evolutionary computation, spiking neural networks.

LE ZHANG received the B.S. degree in automation from the Department of Telecommunications, Lanzhou Jiaotong University, China, in 1998, and the M.S. degree in systems engineering and the Ph.D. degree in control theory and control engineering from Northeastern University, China, in 2005 and 2008, respectively. He is currently a Postdoctoral Fellow with the Dalian University of Technology, and a Domestic Visiting Scholar of Tsinghua University. He is also a Professor and the Scientific Research Dean of the School of Information Engineering, Shenyang University. His current research interests include evolutionary computation, system optimization theory, and applications.

YINGKUI DU received the Ph.D. degree from the Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, in 2010. He is currently an Associate Professor with the Key Laboratory of Equipment Manufacturing Comprehensive Automation, School of Information Engineering, Shenyang University. His current research interests include visual perception technology and the Internet of Things technology.

ZHONGHU YUAN received the B.S. degree from Nankai University, in 1984, and the M.S. and Ph.D. degrees from Northeastern University, in 1989 and 1994, respectively. He has been a Professor with the School of Information Engineering, Shenyang University. He has authored five books, and over 60 articles in international journals and conference proceedings. He has made a deep research on scientific research and teaching in the field of control theory and control engineering, computer software application. He has long been engaged in research on electronic information theory, technology, and application.