Received September 19, 2020, accepted September 30, 2020, date of publication October 12, 2020, date of current version October 22, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3029966

Weighted Fuzzy Production Rule Extraction Using Modified Harmony Search Algorithm and BP Neural Network Framework

HANG-CHENG LI[1], KAI-QING ZHOU[1], LI-PING MO[1], AZLAN MOHD ZAIN[2], (Member, IEEE), AND FENG QIN[2]

[1]College of Information Science and Engineering, Jishou University, Jishou 416000, China
[2]Big Data Centre, Universiti Teknologi Malaysia, Skudai 80310, Malaysia

Corresponding author: Kai-Qing Zhou (jsu_computer@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 62066016, in part by the Natural Science Foundation of Hunan Province, China, under Grant 2020JJ5458 and Grant 2019JJ40234, in part by the Research Foundation of Education Bureau of Hunan Province, China, under Grant 18B317 and Grant 19A414, and in part by the Research-Based Study and Innovative Experimental Project for College Students in Hunan Province under Grant 20180592.

ABSTRACT Compared with rules in the form of 'IF-THEN,' weighted fuzzy production rules (WFPRs) have more robust knowledge expression capabilities, but weighted fuzzy production rules are more difficult to obtain. The weighted fuzzy production rules obtained using traditional neural network methods have shortcomings, such as insufficient precision and insufficient knowledge extraction. Focusing on the mentioned shortages, a modified weighted fuzzy production rules extraction approach is proposed by combining the modified harmony search algorithm, and neural network. The method consists of three main stages. First, a global optimal adaptive harmony search algorithm (AGOHS) is proposed to overcome the traditional harmony search algorithm's existing poor adaptive ability. Then, the AGOHS algorithm is used to optimize the neural network's initial weights to improve the neural network's training efficiency. Finally, extract the WFPRs with IF-THEN from the trained neural network and give the corresponding fuzzy reasoning. Through the WFPRs extraction experiments using IRIS and PIMA data sets reveal the proposed rule extraction framework has some apparent highlights, such as high accuracy, the smaller number of generated rules, and low redundancy.

INDEX TERMS Rule extraction, modified harmony search algorithm, BP neural network framework, weighted fuzzy production rule.

I. INTRODUCTION

Production Rule with 'IF-THEN' formalism is one of the most common representation forms of knowledge in the field of artificial intelligence, which not only has the advantages to understand and to add easily but also to delete or to update related information [1]. To improve the representation ability of traditional production rule, a weighted fuzzy production rule (WFPR) was produced to express vague knowledge [2]. Since then, how to accurately extract WFPR from related data set has been widely discussed by scholars. Andrews *et al.* reviewed the main neural network rule extraction approaches [3]. According to their survey, the neural network rule extraction methods could be divided into two categories, that is, methods by structural analysis and that of using performance analysis [4], [5]. The former includes the Gallant algorithm [6], Subset algorithm [7] and M of the following N(MOFN) algorithm [8], etc., while the latter has Rule from Facts (RF) algorithm, Rule from Networks (RN) algorithm [9] and Benitez algorithm [10], etc.

The Gallant algorithm explains how the neural network concludes a given case by extracting a single rule [6] The Subset algorithm uses a breadth-first search to extract ordinary IF-THEN ruless [7]. Moreover, the technical key of the MOFN algorithm is to quote the concept of equivalence classs [8]. On the other hand, the RF algorithm uses a heuristic search to extract rules directly from the facts after pruning the search spaces [9]. The RN algorithm

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

extracts rules from trained neural networks [9]. Furthermore, the Benitez algorithm proposes that the standard three-layer feedforward neural network learning system and the fuzzy rule-based learning system are equivalents [10]. Likewise, Amit Gupta *et al.* proposed another classification method based on the characteristics of some existing extraction rule algorithms in the ''Input Network Structure Output Extraction Knowledge'' stage. They classified the rule extraction algorithms into two categories, one is the generation and detection method, and the other is the analysis methods [11], [12].

However, the above algorithms have some shortcomings in different degrees, such as insufficient classification accuracy, large calculation amount, poor processing ability for continuous data, and single extraction rule form. Hence, it is necessary to need soft computing technology to optimize performance of related systems [13]–[15]. Some hybrid extraction algorithms were proposed to overcome these shortcomings and achieved the expected goals using evolutionary algorithms and neural networks. For instance, genetic algorithms are used to optimize the neural network to achieve better rules extractions [16], and the ant colony algorithm is used to extract rules from the trained neural network [17]. In recent years, some hybrid swarm intelligence optimization algorithms have been used to optimize neural networks' parameters. For instance, big bang-big crunch (BBBC) optimization and particle swarm optimization (PSO) is applied in the parameter optimization for Interval type-2 fuzzy neural networks (IT2FNNs) [18], and six learning algorithms, including biogeography-based optimization, particle swarm optimization, genetic algorithm, ant colony optimization, evolutionary strategy, and population-based incremental learning are used to explore the best ANN parameter optimization combination [19].

Harmony search algorithm (Harmony search, HS) is an emerging intelligent optimization algorithm [20]. Because its algorithm concept is simple, there are few adjustable parameters, and it is easy to implement, compared with other classic meta-enlightened algorithms (such as genetic algorithm [21], ant colony algorithm [22], particle swarm algorithm [23]). Besides, HS does not require strict mathematical processing of the objective function and the constraint function and can be solved in parallel and distributed for various complex problems, so it can be used to optimize the neural network. For instance, Kulluk *et al.* [24], [25] used different harmony search algorithms to optimize the neural network. The objective function of the harmony search algorithm was set as the objective function of the neural network. The experiment achieved excellent results, mainly by optimizing the initialization weight of the neural network to improve the efficiency of neural network training. The AGOHS algorithm proposed in this paper has good global optimization and self-adaptive capabilities. In theory, it can perform a good error optimization on the objective function of the neural network to improve the knowledge accuracy of the extracted rules.

By inspired by the existing rule extraction approach using NN and Soft computing techniques, this paper proposes a method of extracting weighted fuzzy production rules using hybrid neural networks and improved harmony search algorithm to solve the classification issue of continuous real-valued attributes. The discussed rule extraction approach includes the following three main phases.

1. The improvisation stage of the traditional HS algorithm may cause a temporary stay due to local optimization, which affects the convergence speed and accuracy of the algorithm. Hence, a modified HS algorithm, namely AGOHS, is proposed to overcome the listed shortages above by improving the bandwidth adjustment and the generation mechanism.

2. Then, the AGOHS algorithm is used to optimize the BP neural network's initial weight for improving the training efficiency of the neural network, and structural learning with forgetting of NN is modified by combining the BP algorithm.

3. Finally, the hidden knowledge in the neural network is transformed into WFPRs for extracting the data set. Meanwhile, two classical data_sets are employed to verify the feasibility of the proposed framework.

Compared to the existing approaches, the main contributions of the proposed method could be summarized in the following two aspects:

1. A modified harmony search algorithm, namely AGOHS, is proposed to overcome the existed drawbacks based on the specific improvements below. On the one hand, a novel bandwidth adjustment method is discussed to enhance the bandwidth's adaptive ability for some specific situations. On the other hand, a revised operation using the intra-population difference and the corresponding update mechanism is proposed to random generate harmony variables for increasing the diversity of harmony choices and enhancing the robustness of newly generated harmony in updating the harmony memory.

2. The proposed AGOHS is used to optimize neural networks and implement rule extraction for improving the accuracy of the extracted rules.

The rest parts of this paper are organized below. Section Two reviews the related notions used in this manuscript. Then, a modified HS algorithm and the corresponding performance testing using testing functions are discussed in Section Three. Section Four gives the entire design of the proposed WFPR extraction approach utilizing the modified HS algorithm and BP. Section Five verifies the feasibility of the proposed approach by two data sets. Finally, Section Six recalls the whole manuscript.

II. RELATED NOTIONS

A. BASIC HARMONY SEARCH (HS) ALGORITHM

Harmony Search (HS) algorithm is a novel soft computing technique, proposed by Geem *et al.* in 2001, which is inspired by the underlying principles of the musicians' improvisation of the harmony [20]. The basic HS algorithm can be divided into the following five steps, which are

TABLE 1. Parameters and their corresponding functions.

Parameters	Meanings	Functions
HMS	Harmony memory size	Randomly generate a harmony memory from a solution space, and the HMS controls the size of the harmony memory
HMCR	Harmony memory considering rate	Control the probability of extracting a harmony from the existing population (HM harmony memory)
PAR	Pitch adjusting rate	Control the probability of fine-tuning the extracted harmony
BW	Bandwidth	Control the amplitude of fine-tuning
T_{max}	total iterations	Control the number of total iterations
UB、LB	Search range upper and lower bounds	Control the value range of the solution space

Step 1 (Define the Problem and Parameter Values): Define a minimization problem as in Equation 1 and determine the relevant parameter values (as shown in Table 1).

$$\min f(x), \quad x = \{x_1, x_2, \ldots, x_n\} \in R^n \tag{1}$$

Step 2 (Initialize the Harmony Memory (HM)): Randomly generate $|HMS|$ harmonies $(X^1, X^2, \ldots, X^{HMS})$ from the solution space and put them into HM, and record the corresponding f(x). The form of HM is shown in Equation 2, where f(X) is the function value of harmony.

$$HM = \begin{bmatrix} X^1 \\ X^2 \\ \vdots \\ X^{HMS} \end{bmatrix} \begin{bmatrix} x_1^1 & x_1^2 & \cdots & x_n^1 & |f(X^1) \\ x_1^2 & x_2^2 & \cdots & x_n^2 & |f(X^2) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1^{HMS} & x_2^{HMS} & \cdots & x_n^{HMS} & |f(X^{HMS}) \end{bmatrix} \tag{2}$$

Step 3 (Improvisation (Generating a New Harmony)): A random number r_1 is generated between *[0, 1]* and compared with *HMCR*. If $r_1 < HMCR$, a harmony variable is randomly generated from *HM*; otherwise, a harmony variable is randomly generated from the solution space. If the harmony variable is obtained from HM, then fine-tune the harmony variable. Specifically, a random number r_2 is generated between *[0, 1]*. If $r_2 < PAR$, adjust the obtained harmony variable according to the *BW* to obtain a new harmony variable; otherwise, no adjustments are made; finally, a new harmony x_{new} is obtained.

Step 4 (Update the HM): Evaluate x_{new} to get $f(x_{new})$. If $f(x_{new})$ is better than at least one function value in HM, that is, $f(x_{new}) < f(x_{worst})$, then x_{new} replaces harmony x_{worst}, which has the worst value of the function in HM; otherwise, no modification is made.

Step 5 (Check Whether the Algorithm Is Terminated): Repeat steps 3 and 4 until the number of iterations reaches T_{max}.

B. ERROR BACK PROPAGATION (BP) ALGORITHM

In 1986, Rumelhart *et al.* [26] designed a new type of multi-layer feedforward network, named back propagation neural network (BPNN), whose structure includes input layer (I), hidden layer (H) and output layer (O), and its objective function is demonstrated as equation 3.

$$E = \frac{1}{2}\sum_k (o_k - t_k)^2 \tag{3}$$

BPNN uses error back propagation algorithm to realize network learning and training. The algorithm includes two processes: forward signal propagation and the error back-propagation. Therefore, the error back-propagation algorithm is also called BP algorithm. By learning and training the network through the BP algorithm, a set of connection weights with hidden knowledge, that is, implicit production rules, can be obtained. With these hidden rules, unknown samples can be simulated and predicted.

C. STRUCTURAL LEARNING WITH FORGETTING OF NN ALGORITHM

Structural Learning with Forgetting of NN algorithm was proposed by Masumi *et al.* [27] in 1996 by adding an L1 regularization in the commonly used BP algorithm objective function. During the training process, the weight of unimportant connections is close to 0. After training the network and revising the threshold, some connections less than the threshold are cut off to obtain a simple network structure. The objective function is shown in equation 4.

$$E_f = E + \varepsilon' \sum_{i,j} |w_{ij}| = \frac{1}{2}\sum_k (o_k - t_k)^2 + \varepsilon' \sum_{i,j} |w_{ij}| \tag{4}$$

where $E = \frac{1}{2}\sum_k (o_k - t_k)^2$ is the commonly used BP algorithm objective function, $\varepsilon' \sum_{i,j} |w_{ij}|$ is the L1 regularization, and ε' is the regularization factor. Derived by the formula of BP algorithm, is the amount of change $\Delta w_{ij} = \Delta w'_{ij} - \varepsilon\,\mathrm{sgn}(w_{ij})$ of connection weight in each iteration, $\Delta w'_{ij} = -\eta \frac{\partial E}{\partial w_{ij}}$ is the amount of change of connection weight in BP algorithm, η is the learning rate, $\varepsilon = \eta\varepsilon'$ is the attenuation value of connection weight in each iteration, and $\mathrm{sgn}(w_{ij})$ is the sign function, when w_{ij} is positive, take 1; when negative, take -1.

D. WEIGHTED FUZZY PRODUCTION RULE (WFPR)

Production rule with 'IF-THEN' formalism is common method to represent knowledge. Due to the inaccuracy, the imperfection of data, or inferior methods of acquiring knowledge, the knowledge held by people is probably inaccurate. What is more, fuzziness and uncertainty are two critical aspects of knowledge inaccuracy. To improve the representation ability of the production rule with 'IF-THEN' formalism, fuzzy production rules have been produced to express vague knowledge. In addition, on the basis of fuzzy production rules, the weighted fuzzy production rules further express the weight attributes between the antecedents of the rules and between the rules, which has stronger knowledge representation ability. The form of the combined weighted fuzzy production rules [2] is as follows:

R: ***IF*** V_i is A_i AND ... AND V_n is A_n ***THEN*** U is B, $CF_R, L_{wi}, \ldots, L_{wn}, \lambda_1, \ldots, \lambda_n$, *Gw(R)*

Fact 1: V_1 is A'_1 CF_{F1}, ..., Fact *n*: V_n is A'_n CF_{Fn}

Where $V_1, \ldots, V_n$ and U are attributes, $A_1, \ldots, A_n$ and B are the values of these attributes, and they are all ambiguous. $LW_i(1 \leq i \leq n)$ represents the local weight of antecedent "V_i is A_i", and each LW_i is non-negative; *Gw(R)* represents the global weight of rule R ($Gw(R) \geq 0$). The local weight indicates the relative importance of the rule's antecedents to the rule's conclusion, and global weight is used to indicate the relative importance of each rule's contribution in order to achieve the final goal; λ_i is the threshold of the similarity between the attribute value A_i and the observed value A'_i to limit the activation of the rule, thereby excluding invalid results.

III. ADAPTIVE HARMONY SEARCH UTILIZING GLOBAL OPTIMAL

The improvisation stage may cause a temporary stay due to local optimization, which may affect the algorithm's convergence speed and accuracy. Inspired by [28]–[32], an adaptive harmony search utilizing global optimal (AGOHS) is proposed based on the following aspects. First aims at the adjustment method of the bandwidth of the HS algorithm in the improvisation stage. Secondly, it uses the difference to generate new harmony variables within the population randomly. Finally improves the generation mechanism of new random harmony after the improvisation.

A comparison is executed to verify the feasibly of the AGOHS by 13 classic test functions in two different dimensions using among AGOHS and other three improved HS algorithms. Experimental results reveal the AGOHS has good global search ability and convergence speed.

A. IMPROVEMENT IDEAS

Compared with the traditional HS algorithm, the AGOHS proposed in this paper mainly improves around the improvisation stage, namely:

(1) Improve the bandwidth's adjustment method so that the bandwidth can adapt to specific situations.
(2) In the case that the generated random number is greater than the harmony memory considering rate (HMCR), the operation of randomly generating a harmony variable using intra-population differences is added to improve the diversity of harmony selection.
(3) Improve the generation mechanism of new random harmony after the improvisation stage and enhance the robustness of the newly generated harmony in the update HM stage.

Specific improvement measures are shown in Table 2.

B. AGOHS SPECIFIC STEPS

In the proposed AOGHS framework, it is the current iteration number,T_{max} is number of final iterations,r_1, r_2 are random integers in *[1, 2, 3, ...,HMS]*, r_3 *is a random* integer in *[1, 2, 3, ...,n]*. At the stage of a new mechanism for randomly generating harmony,taking the minimum value x_min_i and the maximum value x_max_i of the *i-th* harmony variable set of all harmony in the current population, each cycle randomly generates a harmony variable cmp_i from [x_min_i, x_max_i], and then compares the function error of *cmp* and x_{new}, and finally takes the one with better effect.

It is not difficult to see from the pseudo-code,The time consumption of this algorithm is mainly in the stage of the new harmony generation mechanism and function error calculation.In the while loop, HM is traversed once and the function error is calculated twice in each loop. Assume n is the optimized function's dimension, HMS is the size of the harmony memory, $T_{\max}$ is the total number of iterations, respectively, the overall time complexity of the AGOHS algorithm is $O(n \times HMS \times T_{\max})$.

C. EXPERIMENT AND RELATED ANALYSIS

To test the effectiveness of AGOHS, this paper uses the 13 classic test functions mentioned in [33]–[35] as the benchmark function. Compared with three improved HS algorithms, such as IDHS [30], HSDM [31], and ID-HS-LDD [32], the optimization results are compared and analyzed under the condition that each of the 13 test function optimization operations is run 50 times separately. Among them, the functions F1-F11 are compared after 5000 iterations in 10 and 30 dimensions respectively, and the functions F12 and F13 are compared after 5000 iterations in a fixed dimension.

TABLE 2. Main modifications of AGOHS.

Improvements	Specific improvement measures
Parameter Settings	1. Newly added parameters $HMCR_{min}$, $HMCR_{max}$, PAR_{min}, PAR_{max} and F. Among them, $HMCR_{min}$ is the minimum harmony memory considering rate, $HMCR_{max}$ is the maximum harmony memory considering rate, PAR_{min} is the minimum pitch adjusting rate, PAR_{max} is the maximum pitch adjusting rate, and F is the harmony factor of the adjustable parameter. 2. The value of the bandwidth (BW) is no longer directly assigned in the parameter setting stage, but is adaptively adjusted according to the specific situation. 3. x_min_i represents the minimum value in the set of the *i-th* harmony variables of all harmony in the population during the HM update stage of the current iteration, x_max_i represents the maximum value in the set of the *i-th* harmony variables of all harmony in the population.
Improvisation	1. When $r_1<HMCR$, fine-tune HM_i^{worst} HM_i^{worst} is the *i-th* of the worst harmony in the current iteration, bandwidth BW= $(HM_i^{best} - HM_i^{worst})$; 2. If $r_2<PAR$, a random harmony variable HM_k^{best} in the harmony with the best iteration effect is kept, and k is an integer randomly selected in *[1, n]*; 3. If $r_1>HMCR$, then $X_{new}(i) = HM_i^{r_1} + r_2 \times (HM_i^{best} - HM_i^{worst})$.
HM's update mechanism	The loop randomly generates a harmony variable cmp_i from $[x\_min_i, x\_max_i]$, and the loop completes to generate a new harmony *cmp*. Compare the function errors of *cmp* and x_{new}. If $f(cmp)<f(x_{new})$, $x_{new}=cmp$; otherwise, $x_{new}=x_{new}$.
HMCR dynamic adjustment	$HMCR(it) = HMCR_{\min} + (HMCR_{\max} - HMCR_{\min}) \times T_i / T_{\max}$
PAR dynamic adjustment	$PAR(it) = PAR_{\max} - (PAR_{\max} - PAR_{\min}) \times T_i / T_{\max}$

(**PS:** ***Ti*** is the number of current iterations)

1) TEST FUNCTION DESCRIPTION

The expressions and characteristics of the 13 classic test functions are as table 3:

2) COMPARISON RESULTS AND ANALYSIS

The characteristics of the selected modified HSs are listed below.

(1) IDHS proposes a novel improvisation scheme based on improved difference, each iteration randomly fine-tunes a harmony value;

(2) HSDM proposes an improvisation scheme based on the adaptive adjustment of differential mutation tones, using random differences between populations to generate BW;

(3) ID-HS-LDD proposes a new harmony generation mechanism.

On the basis of the difference, by introducing a linear dynamic change model, the search field of the optimal value is continuously narrowed to achieve the purpose of algorithm improvement. Moreover,the parameter settings of the four improved HS algorithms are shown in Table 4.

Figures 1-13 is a random set of running results in the experiment. The functions F1-F13 are the convergence curves when the dimensions of each benchmark function are 10 and 30, where (a) is the convergence curve of the four harmony algorithms in dimension 10, (b) is the convergence curve of the four harmony algorithms of dimension 30. Among them, the abscissa is the number of iterations, the ordinate is the optimal harmony function value, the solid red line represents AGOHS, the blue dotted line represents IDHS, the black dotted line represents HSDM, and the green dotted line represents ID-HS-LDD.

It can be seen from Figure 1-13 that the AGOHS algorithm proposed in this paper has faster convergence speed and accuracy than the other three types of improved HS algorithms in most cases.

For unimodal benchmark functions, Sphere, and Rosenbrock function (Figures 1-2), when the dimensions are 10 and 30, AGOHS's early convergence speed and final iteration completion accuracy are superior to the other three algorithms.

TABLE 3. The expressions and characteristics of the 13 classic test functions.

No.	Name	Function	Function type	Search space	$f(x)_{\min}$
F1	Sphere	$f(x)=\sum_{i=1}^{N} x_i^2$	unimodal	[-5.12,5.12]	0
F2	Rosenbrock	$f(x)=\sum_{i=1}^{D\text{-}1}[100(x_{i+1}-x_i^2)+(x_i-1)^2]$	unimodal	[-30,30]	0
F3	Ackley	$f(x)=-20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2})-\exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)+20+e$	multimodal	[-32,32]	0
F4	Griewanks	$f(x)=\frac{1}{4000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})+1$	multimodal	[-600,600]	0
F5	Weierstrass	$f(x)=\sum_{i=1}^{D}(\sum_{k=0}^{k\max}[a^k\cos(2\pi b^k(x_i+0.5))])-D\sum_{k=0}^{k\max}[a^k\cos(2\pi b^k*0.5)]$ (a=0.5,b=3,kmax=20)	multimodal	[-0.5,0.5]	0
F6	Rastrigin	$f(x)=\sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i)+10)$	multimodal	[-5.12,5.12]	0
F7	Noncontinuous Rastrigin	$f(x)=\sum_{i=1}^{D}(y_i-10\cos(2\pi y_i)+10)$ (If $\lvert x_i\rvert<0.5$, then $y_i = x_i$, else $y_i = round(2x_i)/2$.	noncontinuous multimodal	[-5.12,5.12]	0
F8	Schwefel	$f(x)=418.9829\times n-\sum_{i=1}^{n}x_i\sin(\lvert x_i\rvert^{\frac{1}{2}})$	multimodal	[-500,500]	0
F9	Levy	$f(x)=\sin^2(\pi w_1)-\sum_{i=1}^{N-1}(w_i-1)^2[1+10\sin^2(\pi w_i+1)]+(w_N-1)^2[1+\sin^2(2\pi w_D)]$	multimodal	[-10,10]	0
F10	Bohachevsky	$f(x)=\sum_{i=1}^{N-1}[x_i^2+2x_{i+1}^2-0.3\cos(3\pi x_i)-0.4\cos(4\pi x_{i+1})+0.7]$	multimodal	[-15,15]	0
F11	Alpine 1	$f(x)=\sum_{i=1}^{N}\lvert x_i\sin(x_i)+0.1x_i\rvert$	multimodal	[-10,10]	0
F12	Shubert	$f(x)=\prod_{i=1}^{2}\sum_{j=1}^{5}j\cos((j+1)x_i+j)$	fixed-dimenstion multimodal	[-10,10]	-186.73
F13	Trid 10	$f(x)=\sum_{i=1}^{10}(X_i-1)^2-\sum_{i=2}^{10}x_i x_{i-1}$	fixed-dimenstion multimodal	[-100,100]	-210

TABLE 4. Parameter settings of various improved harmony algorithms.

Modified HSs	Parameters
IDHS	*HMS=20, $HMCR_{min}$=0.8, $HMCR_{max}$=0.9, PAR_{min}=0.1, PAR_{max}=0.9, F_2=0.6, F_1=N(0.5,0.3)*
HSDM	*HMS=50, HMCR=0.98*
ID-HS-LDD	*HMS=30, $HMCR_{min}$=0.3, $HMCR_{max}$=0.99, PAR_{min}=0.9, PAR_{max}=0.99*
AGOHS	*HMS=20, $HMCR_{min}$=0.8, $HMCR_{max}$=0.9, PAR_{min}=0.1, PAR_{max}=0.9, F=N(0.5,0.3)*

PS: N*(0.5, 0.3)* means that the standard random decimal with a mean value of 0.5 and a variance of 0.3

For the multimodal benchmark function, Griewanks function (Figure 4), in the case where the dimensions are 10 and 30 dimensions, and the final convergence accuracy is 0, the convergence speed of AGOHS is faster than that of the other three improved algorithms.

For the multimodal benchmark function, Schwefel's function (Figure 8), in the case of a dimension of 10, the other three algorithms all fall into local optimality, causing stagnation. Only AGOHS jumped out of local optimality. The final iteration accuracy is also better than the other three algorithms.

For Shubert function and Trid 10 function with fixed-dimension and optimal value as a fixed negative number (Figure 12-13), the convergence speed and accuracy of AGOHS still have a good effect.

To better understand the experimental results, Table 4 records the experimental comparison results of 50 runs from a numerical point of view, where Dim represents

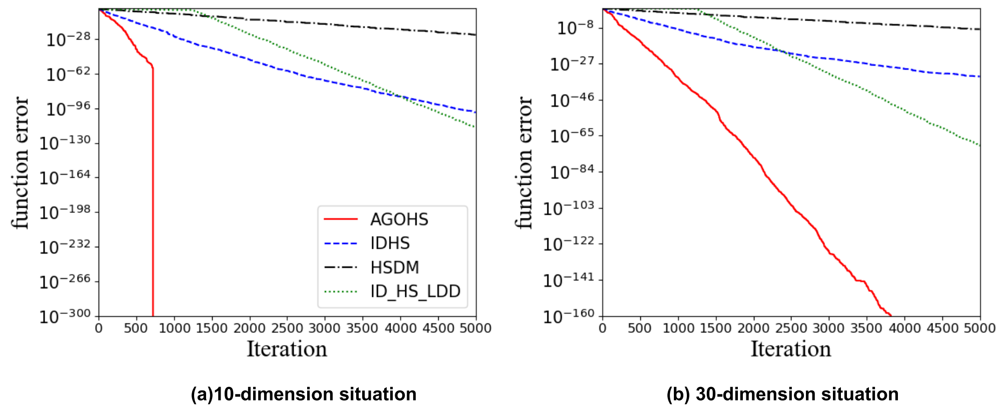


FIGURE 1. Performance comparison on sphere function.

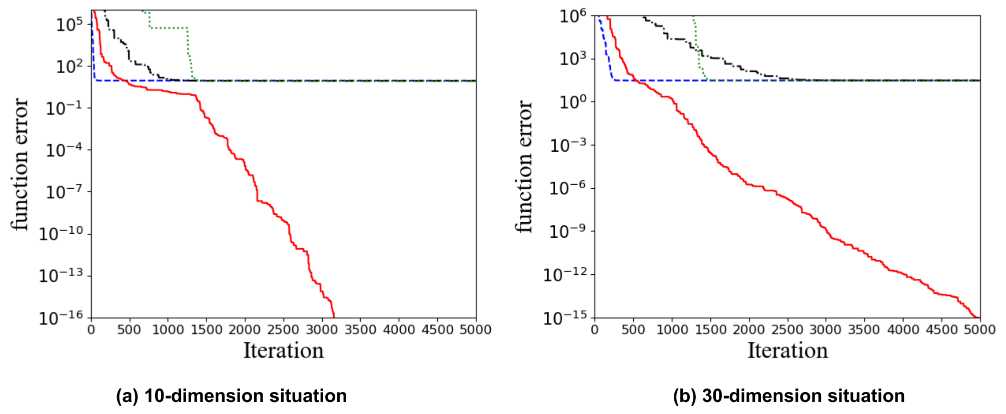


FIGURE 2. Performance comparison on Rosenbrock function.

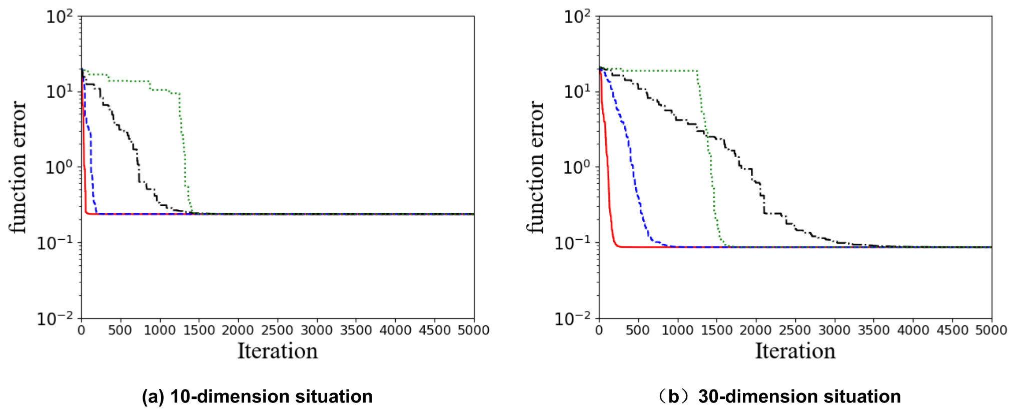


FIGURE 3. Performance comparison on Ackley function.

the dimension; MEAN represents the mean of the optimal harmony function, used to reflect the convergence accuracy of the algorithm; STDV represents the standard deviation of the optimal harmony function value, used to reflect the algorithm's stability, and the value with a good contrast effect is added with black.

It can be seen from Table 5 that, in most cases, the global optimality and adaptability of AGOHS are excellent. For unimodal benchmark functions F1 and F2, When the dimensions are 10 and 30, the resulting MEAN and STDV are both smaller than those of the other three algorithms; for the multimodal benchmark functions F7, F8, and F9, when the

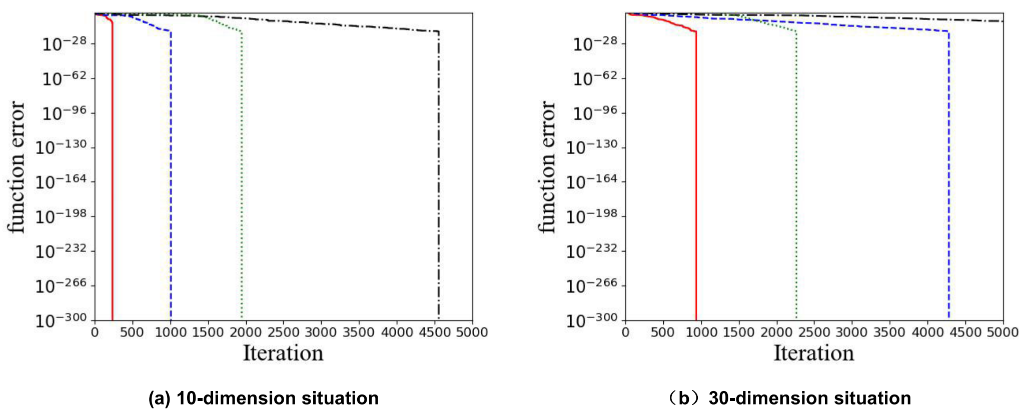


FIGURE 4. Performance comparison on Griewanks function.

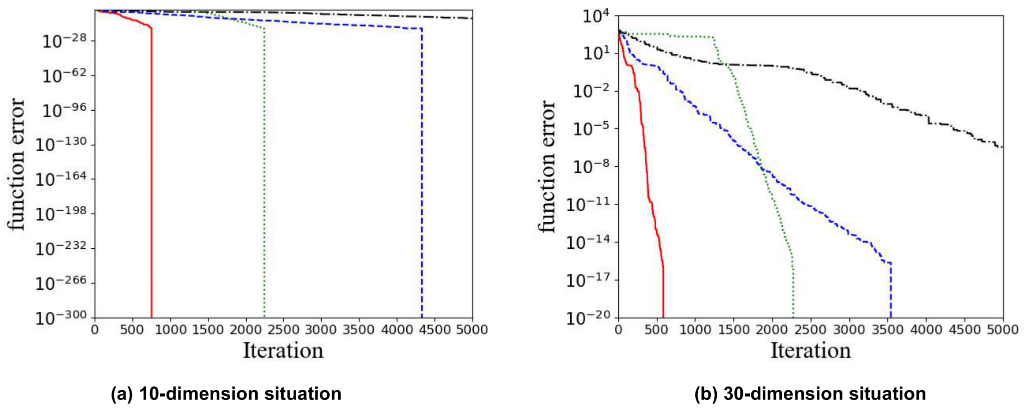


FIGURE 5. Performance comparison on Weierstrass function.

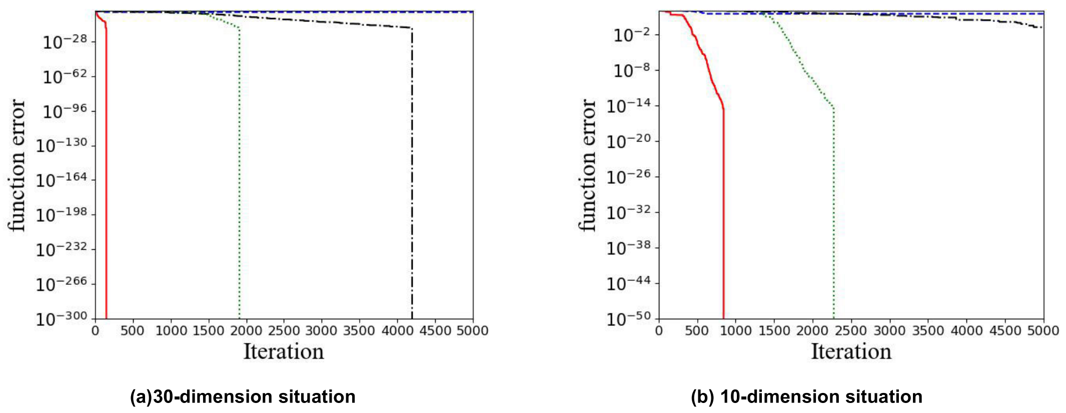


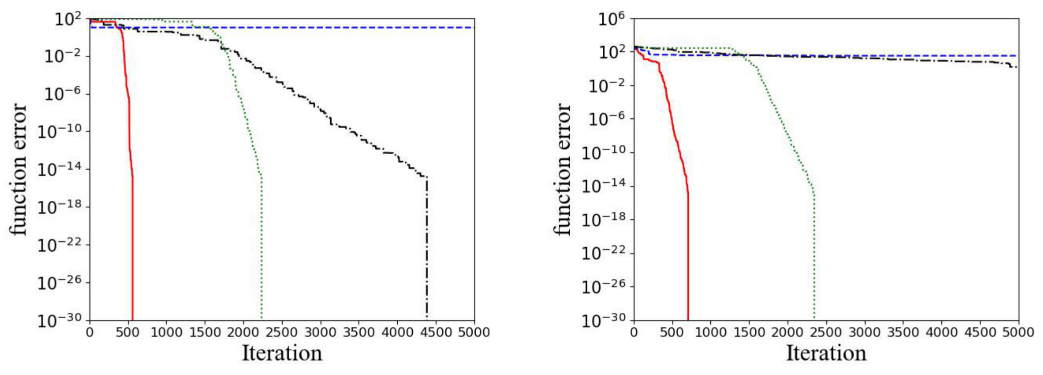
FIGURE 6. Performance comparison on Rastrigin function.

dimensions are 10 and 30, the resulting MEAN and STDV are both smaller than those of the other three algorithms as well.

Based on the analysis results in Figures 1-13 and Table 4, the AGOHS proposed in this paper has better global search capabilities and convergence speed than other improved HS algorithms.

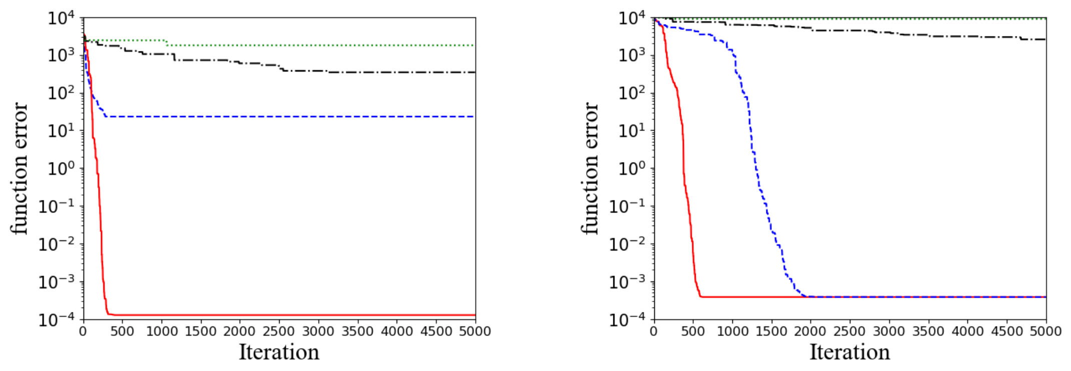
IV. WEIGHTED FUZZY PRODUCTION EXTRACTION BY AGOHS AND BP NEURAL NETWORK

The whole processing of the proposed weighted fuzzy production rule extraction utilizing AGOHS and BPNN in this manuscript could be divided into the following six modules, which are fuzzify the data, determine the neural network topology, use AGOHS to optimize the training of the



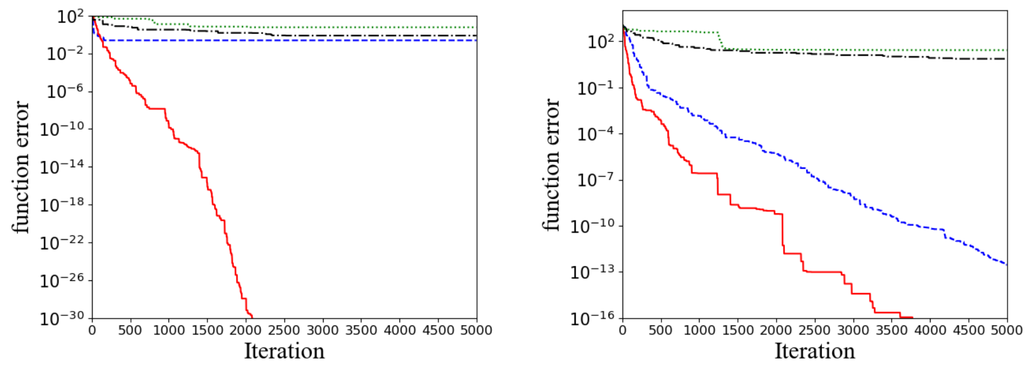
(a)10-dimension situation (b) 30-dimension situation

FIGURE 7. Performance comparison on noncontinuous Rastrigin function.



(a)10-dimension situation (b) 30-dimension situation

FIGURE 8. Performance comparison on Schwefel function.



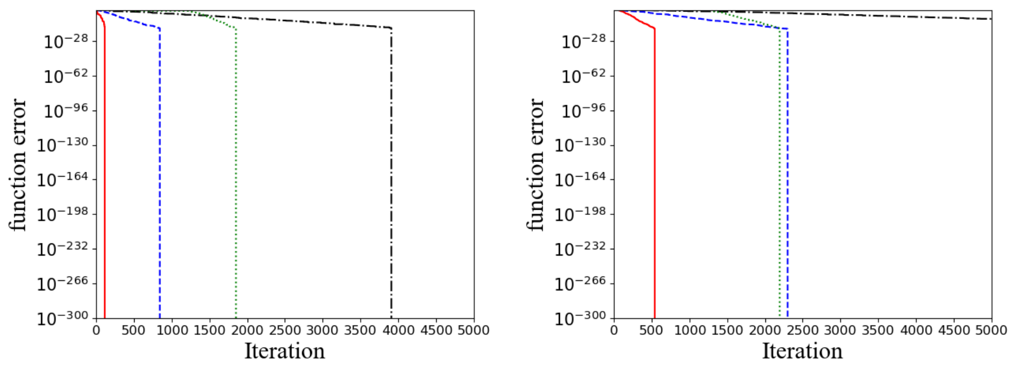
(a) 10-dimension situation (b) 30-dimension situation

FIGURE 9. Performance comparison on Levy function.

neural network, gain the importance index matrix, generate the weighted fuzzy production rules with local weight, and perform fuzzy reasoning on the generated rules. Each module is illustrated in details one-by-one.

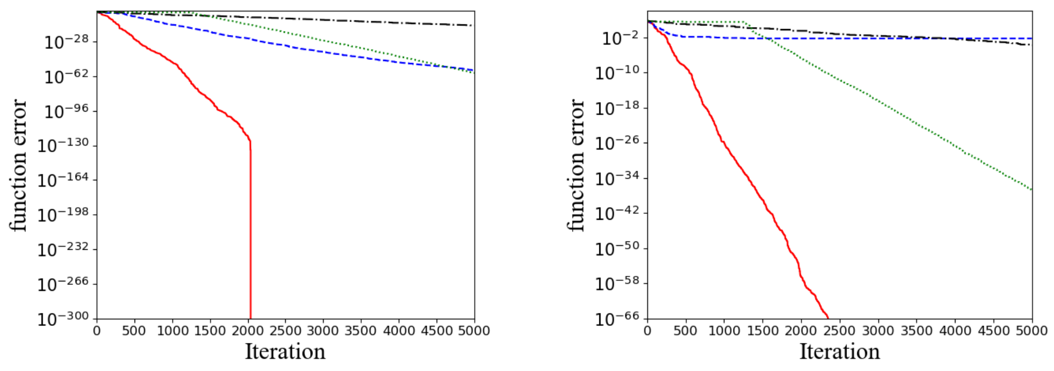
A. DATA FUZZIFICATION

When the data is continuous real values, to obtain the weighted fuzzy production rules of the rule, each attribute value of the training data needs to be divided into semantic values represented by several fuzzy subsets, and the value of each fuzzy subset is fuzzified to between [0, 1]. In order to obtain fuzzy production rules, the attribute values are changed from real type to semantic type before neural network training, mainly to determine the number of semantic values corresponding to each attribute and the membership function corresponding to each semantic value. In terms of



(a) 10-dimension situation (b) 30-dimension situation

FIGURE 10. Performance comparison on Bohachevsky function.



(a) 10-dimension situation (b) 30-dimension situation

FIGURE 11. Performance comparison on Alpine 1 function.

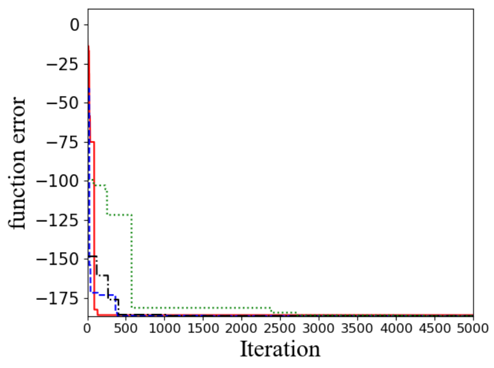


FIGURE 12. Performance comparison on Shubert function.

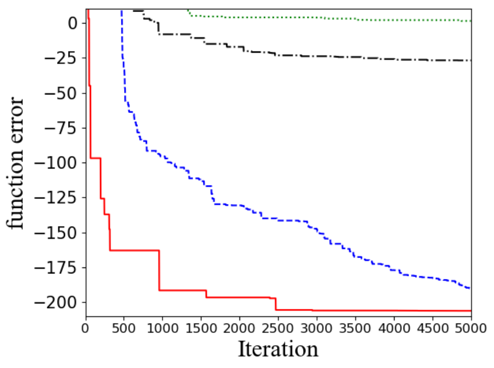


FIGURE 13. Performance comparison on Trid 10 function.

the methods produced, they can be summarized into three categories:

1. Statistical methods, which usually use the related concepts and conclusions of traditional mathematical statistics;
2. Parameter estimation method, which generally estimates the optimal value of a parameter based on a certain theory (such as possibility theory, evidence theory, etc.) under certain optimization criteria;
3. Expert experience method, that is, the membership function used is directly given by domain experts based on experience.

In the data fuzzification of the experiment part of this paper, the IRIS database experiment uses the membership function based on expert experiences, and converts the continuous real value into membership through the membership function; the PIMA database experiment uses fuzzy

TABLE 5. Comparison on experiments' result.

		IDHS	HSDM	ID-HS-LDD	AGOHS
	Dim	MEAN ± STDV	MEAN ± STDV	MEAN ± STDV	MEAN ± STDV
F1	10	4.00E-1001.70E-10 ± 1	1.80E-24 ± 8.60E-25	2.37E-117 ± 1.39E-117	**0.00E+00 ± 0.00E+00**
	30	2.55E-34 ± 2.49E-35	1.01E-09 ± 2.02E-10	5.37E-72 ± 3.30E-72	**4.80E-149 ± 1.78E-146**
F2	10	8.14E+00 ± 5.25E-03	8.50E+00 ± 4.43E-04	4.48E+00 ± 2.89E-03	**1.44E-12 ± 6.79E-14**
	30	2.82E+01 ± 7.26E-05	2.86E+01 ± 2.62E-02	2.87E+01 ± 1.79E-03	**7.19E-12 ± 1.79.E-13**
F3	10	2.36E-01 ± 8.32E-17	4.10E-01 ± 1.12E-11	1.10E-01 ± 8.32E-17	2.36E-01 ± 8.32E-17
	30	8.62E-02 ± 0.00E+00	8.64E-02 ± 3.04E-05	8.62E-02 ± 0.00E+00	8.62E-02 ± 0.00E+00
F4	10	0.00E+00 ± 0.00E+00	1.93E-16 ± 1.17E-16	0.00E+00 ± 0.00E+00	**0.00E+00 ± 0.00E+00**
	30	0.00E+00 ± 0.00E+00	2.58E-06 ± 5.82E-07	0.00E+00 ± 0.00E+00	**0.00E+00 ± 0.00E+00**
F5	10	0.00E+00 ± 0.00E+00	1.31E-13 ± 6.78E-14	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00
	30	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00
F6	10	9.94E+00 ± 3.55E-15	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00	**0.00E+00 ± 0.00E+00**
	30	3.15E+00 ± 4.12E+00	7.17E-02 ± 1.70E-02	0.00E+00 ± 0.00E+00	**0.00E+00 ± 0.00E+00**
F7	10	1.73E-03 ± 1.73E-03	9.47E-04 ± 2.38E-03	0.00E+00 ± 0.00E+00	**0.00E+00 ± 0.00E+00**
	30	3.98E+00 ± 3.76E-04	1.54E+00 ± 0.50E+00	0.00E+00 ± 0.00E+00	**0.00E+00 ± 0.00E+00**
F8	10	1.27E-04 ± 0.00E+00	3.28E+02 ± 4.09E+00	2.69E+03 ± 1.29E+02	**1.27E-04 ± 3.06E-07**
	30	3.81E-04 ± 5.45E-10	3.02E+03 ± 1.44E+02	1.00E+04 ± 2.93E+02	**3.81E-04 ± 5.45E-13**
F9	10	1.29E-32 ± 0.00E+00	4.28E-01 ± 6.42E-02	6.94E+00 ± 2.92E-01	**1.34E-34 ± 0.00E+00**
	30	1.66E-16 ± 4.98E-17	7.79E+00 ± 8.94E-01	2.84E+01 ± 5.87E-01	**1.37E-17 ± 2.11E-19**
F10	10	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00
	30	0.00E+00 ± 0.00E+00	5.04E-07 ± 1.17E-07	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00
F11	10	1.93E-59 ± 3.25E-60	1.43E-12 ± 3.56E-13	7.29E-58 ± 2.79E-58	**0.00E+00 ± 0.00E+00**
	30	9.62E-18 ± 1.85E-18	4.35E-04 ± 5.95E-05	5.57E-38 ± 1.60E-38	**2.65E-75 ± 4.65E-74**
F12	2	-186.73 ± 6.54E-03	-183.78 ± 2.84E-14	-182.00 ± 3.78E+00	**-186.73 ± 9.64E-14**
F13	10	-192.61 ± 1.41E-01	-19.93 ± 1.52E-01	2.05 ± 2.87E-01	**-203.27 ± 4.44E-14**

k-means clustering algorithm to achieve data fuzzification. For example, fuzzy k-means clustering first randomly selects a number of cluster centers, all data points are given a certain degree of fuzzy membership to the cluster centers, and then iteratively modify the clustering center continuously. In the iterative process, the optimization goal is to minimize the distance between all data points and each cluster center and to take the weighted sum of the minimum membership degrees [36]. After determining the number of fuzzy subsets for each attribute, calculate the membership of each fuzzy subset. For example, according to the value of an attribute, it can be divided into three fuzzy subsets of high, medium, and low, and the membership degree of this attribute value in high, middle and low fuzzy subsets is obtained through membership function.

B. DETERMINING NEURAL NETWORK TOPOLOGY AND TRAINING METHODS

In the process of generating a NN framework, the number of input nodes, the number of output nodes, the number of hidden layers, the number of nodes in each hidden layer, and the activation function must first be determined.

In this paper, the three-layer topology of the standard BP neural network is used. Specifically, n represents the number of attributes of the training set. Since each attribute is divided into high, medium, and low fuzzy subsets, the number of input nodes is $3n$. Meanwhile, m is used to indicate the number of types of training set output, and the number of output nodes is m; the number of hidden layer nodes is $\sqrt{3n+m}+\alpha$, and α is an adjustable constant, depending on the specific situation; the activation function uses Sigmoid function.

According to experience, the simple network structure is easy to extract rules. The forgetful structure learning algorithm (SLF) introduced in section 2.3 of this paper can do this better. The algorithm adds a L1 Regularization after the commonly used BP algorithm objective function. During the training process of the BP algorithm, the unimportant connection weight is close to 0, and then a threshold is set. On the premise of ensuring a certain classification accuracy, the connection weight less than this threshold is cropped, so that a neural network with fewer nodes and fewer connections is obtained. Since the goal is to train the neural network connection weights, the bias b is removed during the training process.

Algorithm 1 The Pseudo-Code of the **AGOHS algorithm** Is Described Below.

```
it = 0
/*Initialize the HM*/
for i=1 to HMS do:
    for j=1 to n do:
        $HM_j^i = LB_j + (UB_j - LB_j) \times rand(0, 1)$;
/*Improvise a new harmony*/
While it < $T_{max}$:
    for i=1 to n do:
        if (r<HMCR) then
            $X_{new}(i) = HM_i^{worst} + F \times (HM_i^{best} - HM_i^{worst})$;
            if (r<PAR) then:
                $X_{new}(i) = HM_{r_3}^{best}$;
            end if
        else (i.e. when r>HMCR)
            $X_{new}(i) = HM_i^{r_1} + r_2 \times (HM_i^{best} - HM_i^{worst})$;
        end if
    end for
    /*A new mechanism for randomly generating harmony*/
    for i=1 to $n$ do:
        $x\_min_i = +\inf, x\_max_i = -\inf$;
        for j=1 to HMS do:
            $x\_min_i = \min(x\_min_i, HM_i^j)$;
            $x\_max_i = \max(x\_max_i, HM_i^j)$;
    for i=1 to $n$ do:
        $cmp_i = x\_min_i + (x\_max_i - x\_min_i) \times rand(0, 1)$;
    if $f(cmp) < f(x_{new})$ then
        $x_{new} = cmp$
    if $f(x_{new}) < f(x_{worst})$ then
        $x_{worst} = x_{new}$
    it = it + 1;
    $HMCR(it) = HMCR_{\min} + (HMCR_{\max} - HMCR_{\min}) \times it/T_{\max}$;
    $PAR(it) = PAR_{\max} - (PAR_{\max} - PAR_{\min}) \times it/T_{\max}$;
end while
```

In the simple neural network established and trained by the above method, the connection weight value of some input nodes to hidden nodes is 0, so the change of the input value of these input nodes has no effect on the output value of the neural network, which means that the attribute values represented by these input nodes have no influence on the classification results, and are unnecessary attribute values, so they can be ignored when generating rules, and the attribute value is the feature in the feature set, which plays the role of feature subset extraction.

C. TRAINING OF BP NEURAL NETWORK OPTIMIZATION BY AGOHS

In the traditional BP neural network training, the BP neural network algorithm model has the shortcomings that it is easy to fall into the local optimal and the initial value is relatively random. The selection of the initial value directly affects the training effect of the BP neural network, and a better initial value is beneficial for the BP neural network to jump out of the local optimum, thereby improving the training efficiency. Therefore, the improved HS algorithm is mainly used to optimize the initial value of the BP neural network, so that the BP neural network obtains a better set of initial values. In this paper, the improved HS algorithm (AGOHS) is used to optimize the network weight due to its good global search performance in the early stage of optimization and its strong robustness. The basic idea [24], [25] is to treat the learning process of the network as a process of searching the optimal weight set in the weight space. The calculation process is divided into two parts: the forward propagation of the input signal used to derive the actual output and the backward propagation of the error signal used to correct the weights.

The harmony vectors in HM represent the decision variables in HS, and each vector represents the complete set of connection weights and biases of the BP network. The objective function, that is, the fitness calculation is the minimum formula $E = \frac{1}{2}\sum_k (o_k - t_k)^2$ (i.e., the objective function of the BP neural network). The smaller the sum of the square of the error between the target output and the actual output, the better the individual. Since the values of the network weights are usually in the same range, the same search range is selected for all decision variables [UB, LB]. After determining the minimization objective function, the AGOHS algorithm is used to search, and return to a set of optimal harmony as the initial value of the BP neural network connection weight and threshold after the iteration. Meanwhile, the L1 regularization is added to the training process of the NN to make the connection weights of the trained NN more sparse.

The algorithm flow is illustrated in Figure 14.

D. OBTAINING THE MATRIX OF IMPORTANCE INDEX OF A CERTAIN ATTRIBUTE VALUE TO A CERTAIN CATEGORY

After the neural network is trained, the connection coefficient matrix $input\_hidden_{n\times k}$ from the input layer to the hidden layer and the connection coefficient matrix $hidden\_ouput_{k\times m}$ from the hidden layer to the output layer can be obtained, where n is the number of input layer nodes, k is the number of hidden layer nodes, and m is the number of output layer nodes. Then, the connection weights between the nodes are trimmed, the threshold t is set, and the values less than or equal to t in both matrices are assigned to 0 to obtain $new\_input\_hidden_{n\times k}$ and $new\_hidden\_ouput_{k\times m}$, which realizes the pruning of the neural network; The matrix of importance index $W_{m\times n}$ for each category can be obtained by transposing the matrix multiplied result. Following this, the matrix of importance index $W_{m\times n} = (new\_input\_hidden_{n\times k} * new\_hidden\_ouput_{k\times m})^T$, where the element in the i-th row and j-th column of $W_{m\times n}$

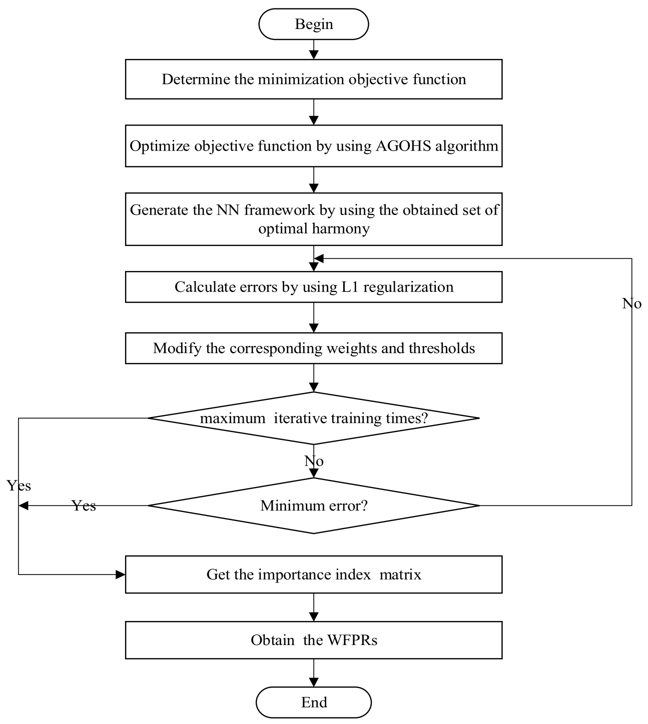

Begin
Determine the minimization objective function
Optimize objective function by using AGOHS algorithm
Generate the NN framework by using the obtained set of optimal harmony
Calculate errors by using L1 regularization
Modify the corresponding weights and thresholds
maximum iterative training times?
No
Yes
Minimum error?
Yes
No
Get the importance index matrix
Obtain the WFPRs
End

FIGURE 14. **The entire flow framework of the extraction processing.**

represents the importance degree of the j-th input node to the i-th output node, because the connection coefficient matrix has been trimmed before, and some of the connection weights are assigned a value of 0, so the matrix of importance index $W_{m\times n}$ for each category obtained by the matrix multiplication will be more sparse. This will make it easy to generate weighted fuzzy production rules in the next step.

E. GENERATE WEIGHTED FUZZY PRODUCTION RULES WITH LOCAL WEIGHT

Through matrix of importance index $W_{m\times n}$, the method of generating the conjunction weighted fuzzy production rule set is as follows:

Each row of the matrix of importance index $W_{m\times n}$ is used to generate one or more rules of the corresponding category of the row: the attribute value corresponding to the element of a row that is not 0 generates the antecedent of the rule, and the corresponding category of the row generates the consequent of the rule. The antecedent of each rule contains only one attribute value of an attribute. If there are multiple elements corresponding to the attribute value of the same attribute that are not 0, multiple rules will be generated, that is, there is no OR in the antecedent; the absolute value of the element will be as the local weight of the corresponding antecedent. If the element is negative, the corresponding attribute value will generate the NOT antecedent. Example 1 is a matrix of importance index:

Example 1:

$$W_{m\times n} = \begin{pmatrix} 3 & -4 & 0 & 0 & -1 & 0 & 2 & 0 & 0 \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \end{pmatrix}$$

The first row corresponds to category B_1. Columns 1, 2, and 3 correspond to the three attribute values A_{11}, A_{12}, A_{13} of the attribute V_1; columns 4, 5, and 6 correspond to the three attribute values A_{21}, A_{22}, A_{23} of the attribute V_2; columns 7, 8, and 9 correspond to the three attribute values A_{31}, A_{32}, A_{33} of the attributeV_3; According to the above method, the first row produces the following two rules:

R_1: IF V_1 is A_{11} [3] AND V_2 is NOT A_{22} [1] AND V_3 is A_{31} [2] THEN U is B_1

R_2: IF V_1 is NOT A_{12} [4] AND V_2 is NOT A_{22} [1] AND V_3 is A_{31} [2] THEN U is B_1

F. CORRESPONDING FUZZY REASONING METHOD

There are many existing methods of fuzzy reasoning, and the same data and rules, due to different reasoning methods, will have very different results. For classification problems, the fuzzy reasoning process can be understood as the process of converting input sample attribute values into corresponding categories using weighted fuzzy production rules. This process is similar to the process of neural network calculation of output values, while the weighted fuzzy production rules are generated based on the weights of the neural network connection, so the reasoning method is generated by simulating the flow of data from the input node to the output node of the neural network. It can also be considered that the weighted fuzzy production rules only contain a part of the hidden knowledge in the network, and the knowledge hidden in the neural network can be converted into weighted fuzzy production rules and corresponding fuzzy reasoning methods.

Based on the above ideas, the constructed reasoning method (based on similarity) is as follows: For each given example.

(1) For each rule R_j in the rule set, calculate the similarity SM_j^i between the antecedent factor A_j^i of the rule and the observed value $A_j'^i$. $A_j'^i$ is a real value, and $SM_j^i = A_j^i(A_j'^i)$ and $A_j^i(x)$ are the membership functions of A_j^i. If the antecedent factor A_j^i has "NOT", $SM_j^i = 1 - A_j^i(A_j'^i)$ represents the similarity of A_j^i with NOT antecedent relative to $A_j'^i$.

(2) Determine which rules are activated by threshold λ_i

(3) The certainty factor of rule R_j:

$$CF_j = \sum_{i=1}^{l} \frac{SM_j^i * LW_j^i}{n_i},$$

n_i is the number of occurrences of the i-th attribute value in the rule set with the same classification result, l is the number of antecedent factors in rule R_j, and LW_j^i represents the local weight.

(4) The output CF_j of the rules of the same category is added to obtain the certainty factor belonging to the category, that is, $CF = \sum_{j}^{J} CF_j$, J indicates the number of rules that conclusion fall into this category.
(5) Compare the size of certainty factors belonging to different categories, and select the category with the largest certainty factor CF as the category to which the sample belongs.

V. EXPERIMENTS AND RELATED ANALYSES

The experiment performed weighted fuzzy production rule extraction on IRIS and PIMA data sets. To be specific, the data set was randomly divided into four parts, three parts were used as the training set, and one was used as the test set. The rule-based fuzzy reasoning extracted from the neural network with and without AGOHS optimization was verified respectively, and the neural network training loss function convergence curve graph optimized with and without AGOHS was compared to verify the feasibility of the proposed extraction algorithm.

The experiments were conducted under the conditions of InTel(R)Core(TM) i5-3470 CPU @ 3.20GHz, 4G memory, Win10 operating system, and the programming language was Python 3.5.

A. IRIS DATABASE EXPERIMENT

There are 150 samples in the IRIS database. Each sample has 4 input attributes: Sepal Length(SL), Sepal Width(SW), Petal Length(PL) and Petal Width(PW), and the unit of measurement is cm. It is divided into three categories: Setosa, Versicolor and Virginia. Each category has 50 samples, and the experimental process is as follows:

To get fuzzy rules, first fuzzify the data. For each attribute, it is represented by 3 fuzzy sets (semantic values): LGR (large), MED (medium) and SM (small). The membership function of the fuzzy set LGR uses the Sigmoid function: $f(x) = \frac{1}{1+\exp(-a_1*(x-c_1))}$; the membership function of the fuzzy set MED uses the bell function: $f(x) = \frac{1}{1+(((x+c_2)/a_2)^2)^{b_2}}$; the membership function of the fuzzy set SM uses the Sigmoid function: $f(x) = \frac{1}{1+\exp(-a_3*(x-c_3))}$. Among them, a and c are the parameters $a_2 = 0.6$, $b_2 = 2, c_2 = -3.25$ control the shape of the curve, which are obtained from the experience of a large number of experiments. In this experiment, the parameter values are $a_1 = 1.5, c_1 = 5.0, a_3 = -1.5, c_3 = 2.0$ respectively.

After the fuzzification process is completed, it is necessary to start to determine the network topology. Each attribute of the sample has 3 semantic attribute values (fuzzy sets). This paper will generate rule antecedents from these 12 semantic attribute values. Since the sample has 12 attribute values and 3 classification results, the neural network established has 12 input nodes in the input layer, 3 nodes in the output layer, 1 hidden layer, and 4 hidden nodes. The hidden layers and nodes are generated based on experience, and the categories are coded as: (1,0,0) indicates the category Setosa, (0,1,0) indicates the category Versicolor, and (0,0,1) indicates the category Virginca. The activation functions of the hidden layer and the output layer are both sigmoid functions. After the network topology is determined, the data set is randomly divided into 4 groups, three of which are used as the training set, and the other is used as the test set, and then the AGOHS is used to optimize the BP neural network method. The parameters of the AGOHS section are set to $HMS=30$, $HMCR_{min}=0.8$, $HMCR_{max}=0.9$, $PAR_{min}=0.1$, $PAR_{max}=0.9$ *and* $F=0.6$, the dimension of each harmony is set to the number of input nodes, namely $D=12$. The neural network is trained using the forgetting structure learning algorithm, using the standard BP algorithm, the maximum number of iterations is 500000, the target error is 1E-4, the learning rate is 0.004, the momentum factor is 0.9, and the penalty factor is 0.001.

After completed this training, majority connection weight values obtained are very close to 0 due to the addition of penalty items. Without changing the classification accuracy, the threshold is set to 0.5, and the connections with the connection weight value lower than this threshold are deleted (i.e., the connection weight value is 0), the matrix of importance index is obtained, $W_{3\times 12}$, as shown at the bottom of the page.

According to the method of generating weighted fuzzy production rules with the local weight from the matrix of importance index, the generated weighted fuzzy production rules are as follows:

The first row of the matrix produces 3 rules classified as Iris-setosa.

IF SL is NOT MED [3.53] **and** PL is SM [26.84] **and** PW is NOT MED [7.59] **THEN** Iris-setosa

IF SL is NOT MED [3.53] **and** PL is NOT MED [4.37] **and** PW is NOT MED [7.59] **THEN** Iris-setosa

IF SL is NOT MED [3.53] **and** PL is NOT LGR [9.8] **and** PW is NOT MED [7.59] **THEN** Iris-setosa

The second row of the matrix produces 3 rules classified as Iris-versicolor.

IF PL is NOT SM [12.3] **and** PW is MED [20.57] **THEN** Iris-versicolor

IF PL is MED [8.49] **and** PW is MED [20.57] **THEN** Iris-versicolor

$$W_{3\times 12} = \begin{pmatrix} 0 & -3.53 & 0 & 0 & 0 & 0 & 26.84 & -4.37 & -9.8 & 0 & -7.59 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -12.3 & 8.49 & -14.08 & 0 & 20.57 & 0 \\ 0 & 2.67 & 0 & 0 & 0 & 0 & -14.38 & -2.69 & 30.33 & 0 & -14.13 & 0 \end{pmatrix}$$

TABLE 6. IRIS comparison of classification accuracy.

	test set	training set
Unoptimized neural network	94.86%	93.30%
The obtained weighted fuzzy production rules and corresponding fuzzy reasoning	93.82%	93.30%
Neural network optimized by AGOHS	99.32%	97.10%
The obtained weighted fuzzy production rules and corresponding fuzzy reasoning	98.42%	96.84%

IF PL is NOT LGR [14.08] **and** PW is MED [20.57] **THEN** Iris-versicolor

The third row of the matrix produces 3 rules classified as Iris-virginica.

IF SL is MED [2.67] **and** PL is NOT SM [14.38] **and** PW is NOT MED [14.13] **THEN** Iris-virginica

IF SL is MED [2.67] **and** PL is NOT MED [2.69] **and** PW is NOT MED [14.13] **THEN** Iris-virginica

IF SL is MED [2.67] **and** PL is LGR [30.33] **and** PW is NOT MED [14.13] **THEN** Iris-virginica

The test set and the training set are used to compare the classification accuracy of the neural network and the extracted rules. The extracted rules are verified by the fuzzy reasoning method given in this paper. The results are shown in the table 6:

B. PIMA DATABASE EXPERIMENT

There are 768 samples in the PIMA database, each sample has 8 attributes, and the values are continuous real numbers, which are divided into two categories. Different from the data fuzzification of the IRIS database experiment, the membership degree of a certain clustering center is obtained by applying the fuzzy K-means clustering method to each attribute. Fuzzy k-means clustering first randomly selects a number of cluster centers, all data points are given a certain degree of fuzzy membership to the cluster centers, and then iteratively revises the cluster centers continuously. In the iterative process, the optimization goal is to minimize the distance from all data points to each cluster center and the weighted sum of membership degrees. The experiment sets 3 center points for each attribute, so that each attribute will have 3 fuzzy attribute values, with A_i representing the i-th attribute, A_{ij} representing the *j-th* fuzzy attribute value of the *i-th* attribute, $i=1, 2, \ldots, 8; j=1, 2$. It is divided into two categories, respectively denoted as CLASS1 and CLASS2. The data set is randomly divided into 4 groups, three of which are used as the training set and the other as the test set.

Except for the different parts of data fuzzification, the methods of other parts are the same as the IRIS database experiment. A three-layer neural network is established. The number of nodes in the input layer is 24, the number of nodes in the hidden layer is 4, the number of nodes in the output layer is 2, and the activation functions are sigmoid functions. The neural network is trained with forgetful structure learning (SLF), and the AGOHS algorithm is added for optimization. The parameters of the AGOHS part are set to HMS=40, $HMCR_{min} = 0.8$, $HMCR_{max} = 0.9$, $PAR_{min} = 0.1$, $PAR_{max} = 0.9$ and $F = 0.6$, and the dimension of each harmony is set to the number of input nodes, that is, $D = 24$. Using standard BP algorithm, the maximum number of iterations is 500,000, the target error is *1E-4*, the learning rate is 0.001, the momentum factor is 0.9, and the penalty factor is 0.002. After completed this training, the threshold is set to 0.4 to trim the network, and the connection weight less than the threshold is set to 0, and then the matrix of importance index is obtained, $W_{2\times24}$, as shown at the bottom of the page.

According to the method of generating weighted fuzzy production rules with the local weight from the matrix of importance index, the generated weighted fuzzy production rules are as follows:

The first row of the matrix produces 6 rules classified as CLASS1.

IF A2 is A21 [10.76] **and** A4 is NOT A43 [1.84] **and** A6 is NOT A61 [2.5] **and** A7 is A73 [2.46] **and** A8 is A83 [7.07] **THEN** CLASS1

IF A2 is A21 [10.76] **and** A4 is NOT A43 [1.84] **and** A6 is NOT A62 [2.21] **and** A7 is A73 [2.46] **and** A8 is A83 [7.07] **THEN** CLASS1

IF A2 is A21 [10.76] **and** A4 is NOT A43 [1.84] **and** A6 is A63 [6.83] **and** A7 is A73 [2.46] **and** A8 is A83 [7.07] **THEN** CLASS1

IF A2 is NOT A22 [17.95] **and** A4 is NOT A43 [1.84] **and** A6 is NOT A61 [2.5] **and** A7 is A73 [2.46] **and** A8 is A83 [7.07] **THEN** CLASS1

IF A2 is NOT A22 [17.95] **and** A4 is NOT A43 [1.84] **and** A6 is NOT A62 [2.21] **and** A7 is A73 [2.46] **and** A8 is A83 [7.07] **THEN** CLASS1

$$W_{2\times24} = \begin{pmatrix} 0 & 0 & 0 & 10.76 & -17.95 & 0 & 0 & 0 & 0 & 0 & 0 & -1.84 & 0 & 0 & 0 & -2.5 & -2.21 & 6.83 & 0 & 0 & 2.46 & 0 & 0 & 7.07 \\ 0 & 0 & 0 & -10.76 & 17.95 & 0 & 0 & 0 & 0 & 0 & 0 & 1.84 & 0 & 0 & 0 & 2.5 & 2.21 & -6.83 & 0 & 0 & -2.46 & 0 & 0 & -7.07 \end{pmatrix}$$

TABLE 7. PIMA comparison of classification accuracy.

	Test set	Training set
Unoptimized neural network	88.86%	86.33%
The obtained weighted fuzzy production rules and corresponding fuzzy reasoning	86.30%	84.25%
Neural network optimized by AGOHS	93.47%	90.83%
The obtained weighted fuzzy production rules and corresponding fuzzy reasoning	91.30%	88.66%

IF A2 is NOT A22 [17.95] **and** A4 is NOT A43 [1.84] **and** A6 is A63 [6.83] **and** A7 is A73 [2.46] **and** A8 is A83 [7.07] **THEN** CLASS1

The second row of the matrix produces 6 rules classified as CLASS2.

IF A2 is NOT A21 [10.76] **and** A4 is A43 [1.84] **and** A6 is A61 [2.5] **and** A7 is NOT A73 [2.46] **and** A8 is NOT A83 [7.07] **THEN** CLASS2

IF A2 is NOT A21 [10.76] **and** A4 is A43 [1.84] **and**A6 is A62 [2.21] **and** A7 is NOT A73 [2.46] **and** A8 is NOT A83 [7.07] **THEN** CLASS2

IF A2 is NOT A21 [10.76] **and** A4 is A43 [1.84] **and** A6 is NOT A63 [6.83] **and** A7 is NOT A73 [2.46] **and** A8 is NOT A83 [7.07] **THEN** CLASS2

IF A2 is A22 [17.95] **and** A4 is A43 [1.84] **and** A6 is A61 [2.5] **and** A7 is NOT A73 [2.46] **and** A8 is NOT A83 [7.07] **THEN** CLASS2

IF A2 is A22 [17.95] **and** A4 is A43 [1.84] **and** A6 is A62 [2.21] **and** A7 is NOT A73 [2.46] **and** A8 is NOT A83 [7.07] **THEN** CLASS2

IF A2 is A22 [17.95] **and** A4 is A43 [1.84] **and** A6 is NOT A63 [6.83] **and** A7 is NOT A73 [2.46] **and** A8 is NOT A83 [7.07] **THEN** CLASS2

The test set and the training set are used to compare the classification accuracy of the neural network and the extracted rules. The extracted rules are verified by the fuzzy reasoning method given in this paper. The results are shown in the table 7:

Through the analysis of the above two data sets, under the premise of ensuring accuracy, the generated weighted fuzzy production rules generate fewer rules and will not generate a great deal of rule redundancy. For the two data sets, the neural network optimized with AGOHS has significantly higher classification accuracy than that without optimization, and the accuracy of the weighted fuzzy production rules obtained is also higher, which verifies the feasibility of using AGOHS to optimize neural network training. From the experimental results, the classification accuracy obtained by the fuzzy reasoning method proposed in this paper is very close to the classification accuracy of the neural network, which shows that the obtained rule set and the corresponding fuzzy reasoning method better contain the knowledge contained in the neural network. From the perspective of algorithm flow, the proposed rule extraction method flow is more clear, the calculation amount is also smaller, and it is more adaptable to the continuous real value data set.

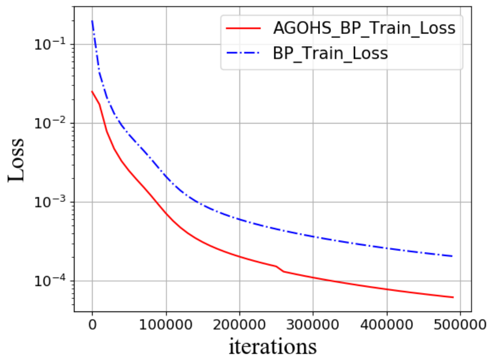


FIGURE 15. Performance comparison on IRIS database's AGOHS_BP and BP.

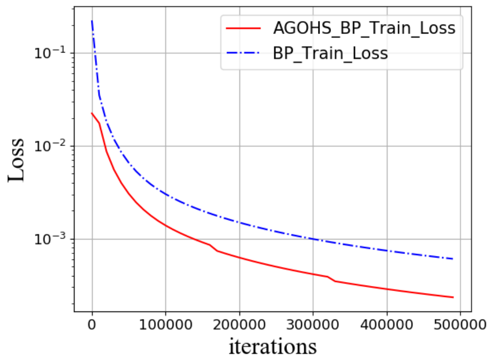


FIGURE 16. Performance comparison on PIMA database's AGOHS_BP and BP.

C. COMPARISON OF NEURAL NETWORK LOSS FUNCTION CURVES

Using the same training set, the Figures 15 and 16 show the convergence curve of the loss function value of the BP neural network with optimized by AGOHS (AGOHS_BP_Train_Loss) and without optimization (BP_Train_Loss) during the training process of the IRIS data set and the PIMA data set, respectively.

The red solid line represents AGOHS_BP_Train_Loss, and the blue dashed line represents BP_Train_Loss. It can be seen from the figures that at the beginning of the iteration, AGOHS_BP_Train_Loss is smaller than BP_Train_Loss. This is because AGOHS has optimized the initialization weights of the neural network. At the end of the iteration, the loss function value of AGOHS_BP_Train_Loss is still smaller than that of BP_Train_Loss, which shows that AGOHS has played an optimal role in the training of neural networks.

VI. CONCLUSION

Focusing on the shortcomings of the traditional weighted fuzzy production rules extraction system using NN, such as insufficient precision and insufficient knowledge extraction, this paper proposes a hybrid weighted fuzzy production rule extraction framework by combining modified HS and BPNN. The entire process of this article could be divided into two parts. First, a modified HS algorithm AGOHS is proposed, which time complexity is $O(n \times HMS \times T_{\max})$, the experimental results of 13 benchmark functions have shown its good global optimization and adaptive capabilities. Second, base on the good global optimization and adaptive capabilities of AGOHS, AGOHS is used to optimize the parameters of the neural network, thereby improving the accuracy and expressive ability of the extracted rules. The experimental results of two databases show that the neural network optimized with AGOHS has higher classification accuracy than without optimization. The accuracy of the resulting weighted fuzzy production rules is also higher, verifying the feasibility of using AGOHS to optimize neural network rule extraction.

REFERENCES

[1] R. R. Yager, "On the interpretation of fuzzy if then rules," *Int. J. Speech Technol.*, vol. 6, no. 2, pp. 141–151, Apr. 1996.

[2] D. S. Yeung and E. C. C. Tsang, "Weighted fuzzy production rules," *Fuzzy Sets Syst.*, vol. 88, no. 3, pp. 299–313, Jun. 1997.

[3] R. Andrews, J. Diederich, and A. B. Tickle, "Survey and critique of techniques for extracting rules from trained artificial neural networks," *Knowl.-Based Syst.*, vol. 8, no. 6, pp. 373–389, Dec. 1995.

[4] M. Chakraborty, S. K. Biswas, and B. Purkayastha, "Rule extraction from neural network using input data ranges recursively," *New Gener. Comput.*, vol. 37, no. 1, pp. 67–96, Jan. 2019.

[5] Z. H. Zhou and S. F. Che, "Rule extraction from neural networks," (in Chinese), *J. Comput. Res. Develop.*, vol. 4, pp. 398–405, Apr. 2002.

[6] S. I. Gallant, "Connectionist expert systems," *Commun. ACM*, vol. 31, no. 2, pp. 152–169, Feb. 1988.

[7] L. Fu, "Rule learning by searching on adapted nets," in *Proc. 9th Nat. Conf. Artif. Intell.*, Anaheim, CA, USA, vol. 2, Jul. 1991, pp. 590–595.

[8] G. G. Towell and J. W. Shavlik, "Extracting refined rules from knowledge-based neural networks," *Mach. Learn.*, vol. 13, no. 1, pp. 71–101, Oct. 1993.

[9] K. Saito, "Rule extraction from facts and neural networks," in *Proc. INNC PARIS*, 1990, pp. 379–382.

[10] J. M. Benitez, J. L. Castro, and I. Requena, "Are artificial neural networks black boxes?" *IEEE Trans. Neural Netw.*, vol. 8, no. 5, pp. 1156–1164, Sep. 1997.

[11] A. Gupta, S. Park, and S. M. Lam, "Generalized analytic rule extraction for feedforward neural networks," *IEEE Trans. Knowl. Data Eng.*, vol. 11, no. 6, pp. 985–991, Nov. 1999.

[12] S. K. Biswas, M. Chakraborty, B. Purkayastha, P. Roy, and D. M. Thounaojam, "Rule extraction from training data using neural network," *Int. J. Artif. Intell. Tools*, vol. 26, no. 03, Jun. 2017, Art. no. 1750006.

[13] Z. Cai and W. Zhu, "Feature selection for multi-label classification using neighborhood preservation," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 1, pp. 320–330, Jan. 2018.

[14] X. Luo, Z. Liu, S. Li, M. Shang, and Z. Wang, "A fast non-negative latent factor model based on generalized momentum method," *IEEE Trans. Syst., Man, Cybern. Syst.*, early access, Nov. 21, 2018, doi: 10.1109/TSMC.2018.2875452.

[15] X. Luo, M. Zhou, Y. Xia, Q. Zhu, A. C. Ammari, and A. Alabdulwahab, "Generating highly accurate predictions for missing QoS data via aggregating nonnegative latent factor models," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 3, pp. 524–537, Mar. 2016.

[16] M. H. Mohamed, "Rules extraction from constructively trained neural networks based on genetic algorithms," *Neurocomputing*, vol. 74, no. 17, pp. 3180–3192, Oct. 2011.

[17] L. Özbakir, A. Baykasoğlu, S. Kulluk, and H. Yapıcı, "TACO-miner: An ant colony based algorithm for rule extraction from trained neural networks," *Expert Syst. Appl.*, vol. 36, no. 10, pp. 12295–12305, Dec. 2009.

[18] J. Wang and T. Kumbasar, "Parameter optimization of interval type-2 fuzzy neural networks based on PSO and BBBC methods," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 1, pp. 247–257, Jan. 2019.

[19] 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 Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 2, pp. 601–614, Feb. 2019.

[20] Z. Woo Geem, J. Hoon Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: Harmony search," *Simulation*, vol. 76, no. 2, pp. 60–68, Feb. 2001.

[21] G. T. Reddy, M. P. K. Reddy, K. Lakshmanna, D. S. Rajput, R. Kaluri, and G. Srivastava, "Hybrid genetic algorithm and a fuzzy logic classifier for heart disease diagnosis," *Evol. Intell.*, vol. 13, no. 2, pp. 185–196, Jun. 2020.

[22] J. Liu, J. Yang, H. Liu, X. Tian, and M. Gao, "An improved ant colony algorithm for robot path planning," *Soft Comput.*, vol. 21, no. 19, pp. 5829–5839, Oct. 2017.

[23] D. Wang, D. Tan, and L. Liu, "Particle swarm optimization algorithm: An overview," *Soft Comput.*, vol. 22, no. 2, pp. 387–408, Jan. 2018.

[24] S. Kulluk, L. Ozbakir, and A. Baykasoglu, "Training neural networks with harmony search algorithms for classification problems," *Eng. Appl. Artif. Intell.*, vol. 25, no. 1, pp. 11–19, Feb. 2012.

[25] S. Kulluk, L. Ozbakir, and A. Baykasoglu, "Self-adaptive global best harmony search algorithm for training neural networks," *Procedia Comput. Sci.*, vol. 3, pp. 282–286, Jan. 2011.

[26] D. E. Rumelhart, G. E. Hinton, and R. Williams, "Learning representations by back propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[27] M. Manngård, J. Kronqvist, and J. M. Böling, "Structural learning in artificial neural networks using sparse optimization," *Neurocomputing*, vol. 272, pp. 660–667, Jan. 2018.

[28] Q.-K. Pan, P. N. Suganthan, M. F. Tasgetiren, and J. J. Liang, "A self-adaptive global best harmony search algorithm for continuous optimization problems," *Appl. Math. Comput.*, vol. 216, no. 3, pp. 830–848, Apr. 2010.

[29] D. Zou, L. Gao, S. Li, J. Wu, and X. Wang, "A novel global harmony search algorithm for task assignment problem," *J. Syst. Softw.*, vol. 83, no. 10, pp. 1678–1688, Oct. 2010.

[30] L. Wang, H. Hu, R. Liu, and X. Zhou, "An improved differential harmony search algorithm for function optimization problems," *Soft Comput.*, vol. 23, no. 13, pp. 4827–4852, Jul. 2019.

[31] A. K. Qin and F. Forbes, "Harmony search with differential mutation based pitch adjustment," in *Proc. 13th Annu. Conf. Genetic Evol. Comput. - GECCO*, 2011, pp. 545–552.

[32] Q. Zhu, X. Tang, Y. Li, and M. O. Yeboah, "An improved differential-based harmony search algorithm with linear dynamic domain," *Knowl.-Based Syst.*, vol. 187, Jan. 2020, Art. no. 104809.

[33] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016.

[34] Y. Zhu, W. Xu, G. Luo, H. Wang, J. Yang, and W. Lu, "Random forest enhancement using improved artificial fish swarm for the medial knee contact force prediction," *Artif. Intell. Med.*, vol. 103, Mar. 2020, Art. no. 101811.

[35] H. Li, P.-C. Shih, X. Zhou, C. Ye, and L. Huang, "An improved novel global harmony search algorithm based on selective acceptance," *Appl. Sci.*, vol. 10, no. 6, p. 1910, Mar. 2020.

[36] J. Zhu, Z. Jiang, G. D. Evangelidis, C. Zhang, S. Pang, and Z. Li, "Efficient registration of multi-view point sets by K-means clustering," *Inf. Sci.*, vol. 488, pp. 205–218, Jul. 2019.

HANG-CHENG LI was born in Yongzhou, China, in 1998. He is currently pursuing the bachelor's degree in computer science and technology with Jishou University. His main research interests include knowledge graph and soft computing techniques.

KAI-QING ZHOU was born in Changsha, China, in 1984. He received the B.S. degree in computer science and technology from Jishou University in 2006, the M.S. degree in computer applied techniques from the Changsha University of Science and Technology in 2011, and the Ph.D. degree in computer science from Universiti Teknologi Malaysia in 2016. He was a Postdoctoral Fellow of the College of Information and Engineering, Central South University, from 2016 to 2018. He is currently an Associate Professor with the Department of Data Science and Big Data Technology, College of Information and Engineering, Jishou University. His main research interests include the fuzzy Petri nets and its applications, the Chinese information process, image processing, and soft computing techniques.

LI-PING MO was born in Yiyang, China, in 1972. She received the M.S. degree in computer application technology from Central South University in 2006. She is a Senior Lab Master with the Department of Computer Science and Technology, College of Information Science and Engineering, Jishou University. Her research interests include Chinese information processing, Petri nets, and related applications.

AZLAN MOHD ZAIN (Member, IEEE) was born in Pahang, Malaysia, in 1974. He received the Ph.D. degree in computer science from Universiti Teknologi Malaysia (UTM), in 2010. He was appointed as the Director of the Big Data Center, UTM, in April 2020. He is currently an Associate Professor with the School of Computing, Faculty of Engineering, UTM. His main research interests include artificial intelligence, modeling and optimization, machining, and statistical process control.

FENG QIN was born in Changsha, China, in 1994. He received the B.E. degree in computer science and technology from Jishou University, in 2018, and the M.E. degree in human–computer interaction from the University of Nottingham, in 2019. He is currently pursuing the Ph.D. degree in computer science with Universiti Teknologi Malaysia. His research interests include fuzzy Petri nets and its applications, and knowledge graph.