International Journal of Cognitive Informatics and Natural Intelligence
Volume 16 • Issue 1

A Hybrid Learning Particle Swarm Optimization With Fuzzy Logic for Sentiment Classification Problems

Jiyuan Wang, Jiangxi University of Science and Technology, China
Kaiyue Wang, Jiangxi University of Science and Technology, China
Xiangfang Yan, Jiangxi University of Science and Technology, China*
Chanjuan Wang, Jiangxi University of Science and Technology, China

ABSTRACT

Methods based on deep learning have great utility in the current field of sentiment classification. To better optimize the setting of hyper-parameters in deep learning, a hybrid learning particle swarm optimization with fuzzy logic (HLPSO-FL) is proposed in this paper. Hybrid learning strategies are divided into mainstream learning strategies and random learning strategies. The mainstream learning strategy is to define the mainstream particles in the cluster and build a scale-free network through the mainstream particles. The random learning strategy makes full use of historical information and speeds up the convergence of the algorithm. Furthermore, fuzzy logic is used to control algorithm parameters to balance algorithm exploration and exploration performance. HLPSO-FL has completed comparison experiments on benchmark functions and real sentiment classification problems respectively. The experimental results show that HLPSO-FL can effectively complete the hyperparameter optimization of sentiment classification problem in deep learning and has strong convergence.

KEYWORDS

Deep Learning, Fuzzy Logic, Particle Swarm Optimization, Scale-Free Network, Sentiment Analysis

INTRODUCTION

An optimization problem refers to finding a set of parameter values in the feasible solution set so that the objective value of the problem can reach the maximum or minimum under some constraints. Various optimization problems in life have prompted the progress of algorithm research. Intelligent optimization algorithms have been vigorously developed in this regard. At present, the utilization of information flow in the network has become a hot topic, and the research on sentiment analysis of large-scale data in the network has gradually attracted the attention of scholars. Nowadays, the most commonly used sentiment analysis method is the text sentiment analysis method based on deep learning. However, due to the large number of hyperparameters involved in this method, an appropriate optimization method is needed to improve the efficiency of this sentiment analysis method.

A Particle Swarm Optimization algorithm (PSO) is a class of classical swarm intelligence optimization algorithms (Kennedy & Eberhart, 1995). A PSO algorithm has a simple principle and fast convergence speed. It has undergone extensive research and development in recent years in many

DOI: 10.4018/IJCINI.314782 *Corresponding Author

fields, such as resource transportation (Singh & Singh, 2023), system reliability tests (Li et al., 2022), high-speed train head shape optimization (He et al., 2022), etc. However, the PSO algorithm, like other swarm intelligence optimization methods, tends to premature maturity and falls into the local optimum (Ding & Gu, 2020). For this reason, many PSO variants have been proposed to enhance the search performance of a PSO. Common variants include using parameter adaptive control to optimize the execution effect of the algorithm, mixing a PSO algorithm with other algorithms or strategies to improve the performance of the algorithm, using different neighborhood structure to optimize the search ability of the algorithm, using multiple population mechanisms to optimize the efficiency of population information interaction, and changing the learning mechanism of particles to improve the performance.

The PSO variants proposed above will be optimized for different characteristics according to their respective algorithm strategies. They are highly dependent on the problem and are not suitable for solving sentiment classification problems. In this study, a hybrid learning particle swarm optimization with a fuzzy logic (HLPSO-FL) algorithm is proposed for sentiment analysis in the context of deep learning. The main contributions of this paper are as follows:

1. Based on the scale-free network topology, a mainstream learning strategy is proposed to reduce the speed of algorithm information transmission and avoid the algorithm falling into the local optimum.
2. Utilizing a random learning method as opposed to the self-learning strategy enhances the population's diversity and prevents its early convergence.
3. According to the state of each particle, fuzzy logic is introduced to dynamically control parameters, such as inertia weight and individual learning factor of each particle, and dynamically adjust the exploration and development capabilities.

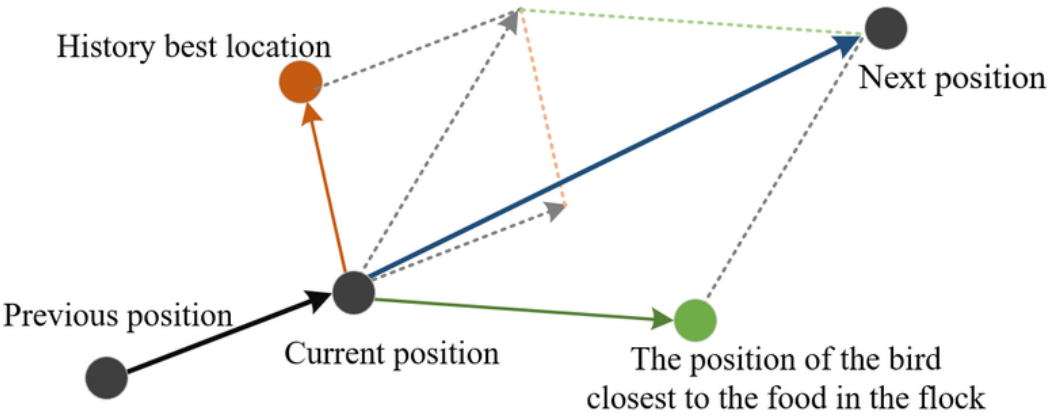
The rest of the paper is structured as follows: The related work of particle swarm optimization algorithm, scale-free network, and sentiment analysis approach is introduced in Related Works. Hybrid learning particle swarm optimization with fuzzy logic details the algorithms outlined in this study. The experimental analysis is arranged in Experimental Results and Analysis. Finally, the summary and prospect of the work of this paper are given.

RELATED WORKS

PSO Algorithm

The standard PSO was proposed by Dr. Eberhart and Kennedy in 1995 (Kennedy & Eberhart, 1995). Its basic idea is derived from the collective behavior of birds when they are foraging: Suppose that the food source is stored somewhere in space, and the birds do not know the location of the food source. The amount of food varies with the distance from the source, and the purpose of the flock is to find the best food source. In the whole search process, they constantly transmit their information to each other and constantly adjust their speed and position through other birds in the group and their information. Finally, the entire flock can be close to food sources. Figure 1 shows the vector diagram of the flock foraging displacement.

Figure 1. Vector diagram of bird flock foraging displacement



In PSO, each particle is a potential solution to an optimization problem, the particle flies in the feasible search space, and continuously adjusts its speed and position according to its speed and the position of other particles. The whole particle swarm gradually approaches the optimal position. Finally, the approximate optimal solution to the optimization problem is obtained.

The position and velocity of the i th particle are denoted as $\left\{X_i^t\left(1\right),X_i^t\left(2\right),\cdots,X_i^t\left(D\right)\right\}$ and $\left\{V_i^t\left(1\right),V_i^t\left(2\right),\cdots,V_i^t\left(\mathrm{D}\right)\right\}$, respectively, where $i=1,2,\ldots,N$, where N is the number of particles in the population, and D is the dimension of the search space. The updated equations for the velocity and position of particle *i* are as follows:

$$V_i^{t+1}=\omega\left(t\right)V_i^t+c_1\cdot r_1\left(Pbest_i^t-X_i^t\right)+c_2\cdot r_2\left(Gbest^t-X_i^t\right) \tag{1}$$

$$X_i^{t+1}=X_i^t+V_i^{t+1} \tag{2}$$

where ω is the inertia weight, r_1 and r_2 are two random numbers uniformly distributed in [0, 1], and c_1 and c_2 are the individual learning factor and social learning factor, respectively. The individual history optimal $Pbest_i^t$ and the global optimal $Gbest^t$ are defined as follows:

$$Pbest_i^t=argmin\left\{f\left(X_i^1\right),\cdots,f\left(X_i^t\right)\right\} \tag{3}$$

$$Gbest^t=argmin\left\{f\left(Pbest_1^t\right),\cdots,f\left(Pbest_N^t\right)\right\} \tag{4}$$

PSO Variants

Due to the poor convergence of standard particle swarm optimization algorithms in practical applications, it is difficult to find a satisfactory feasible solution in a given time range. Many improvement strategies are used to optimize the PSO algorithm, which can be roughly divided into the following five categories:

1. The first category follows parameters adaptive control strategy to optimize the performance of the algorithm. To improve PSO's ability to search, Gupta et al. (2017) used four adaptive inertia weight approaches. To enhance each particle's exploration and exploitation features, Tanweer et al. (2016) developed a self-adjusting inertial weight to govern the dynamics of each particle. A chaos-based inertia weight, which Liu et al. (2020) proposed, is a nonlinear and highly volatile kind of value taking that swings throughout the entire iteration process.
2. Mixing the PSO algorithm with other optimization algorithms or strategies can be classified as the second category of PSO variants. It is expected that the characteristics of the two algorithms will be mixed to improve the ability of the algorithm through this combined implementation. Wang et al. (2019) proposed a self-adaptive mutation differential evolution algorithm based on particle swarm optimization (DEPSO), which adds a DE mutation strategy to PSO to enhance its global exploration ability and avoid premature convergence of the population.
3. The third category of PSO variants alleviates premature convergence by introducing new neighborhood typologies to enhance population diversity. Lin et al. (2019) presented a global genetic learning particle swarm optimization with diversity enhancement by ring topology. Ring topology and global learning components with linearly tuned control parameters are combined with a genetic learning PSO to enhance its diversity, exploration, and adaptability. Xia et al. (2018) used a dynamical topology for a multiswarm particle swarm, periodically reducing the number of subgroups to balance exploration and exploitation capabilities.
4. The construction of a multiswarm particle swarm optimization algorithm by exchanging information between different groups belongs to the fourth category. A novel, dynamic, multiswarm PSO is introduced and discussed by Liang & Suganthan (2005), in which the population is divided into many subswarms. These swarms are frequently regrouped to exchange information among the swarms. The behavior is used to obtain better performance on a complex multimodal optimization problem.
5. The fifth category of PSO variants mainly changes the learning strategy of the PSO algorithm. Cheng & Jin (2015) proposed a social learning PSO (SLPSO) in which each particle learned from a better one than it and the mean behavior of all the particles in the current population. Liang & Suganthan (2005) presented the complete learning PSO (CLPSO), which modifies each particle's velocity using the past optimal data from all other particles (Cao et al., 2018).

Scale-free Network

In 1999, Barabási and Albert found through their research on the World Wide Web that the connectivity of network nodes tends to be a "two-eight distribution", that is, only a few network nodes have a large number of accesses and can be connected to most network nodes in the network, while most network nodes can only connect to a few network nodes. They called complex networks that fit these characteristics "scale-free networks," and then proposed the famous Barabási-Albert (BA) scale-free model (Barabasi & Albert, 1999).

The two main mechanisms used in the BA model are the growth of the network and the preferential connection of nodes, which are constructed as follows:

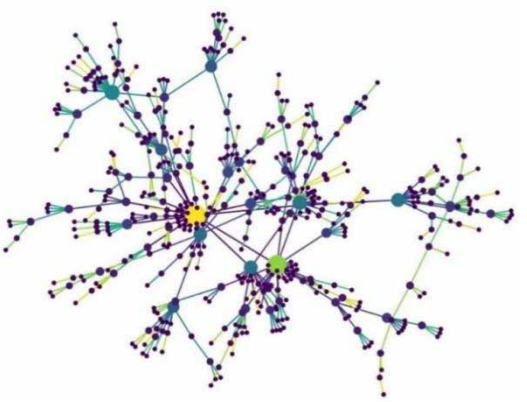
1. Growth: Start with a connected network consisting of m_0 nodes, adding one node at a time.

2. Priority connection: The new node is connected to m existing nodes ($m < m_0$), as shown in Equation (5), according to the degree k_i of node i , which determine the connection probability p_i between the new node and node i .

$$p_i = \frac{k_i}{\sum_j^m k_j} \tag{5}$$

Figure 2 is a schematic diagram of a BA network simulation with a network scale of 500. From the figure, the characteristics of the "two-eight distribution" of the BA network model can be seen.

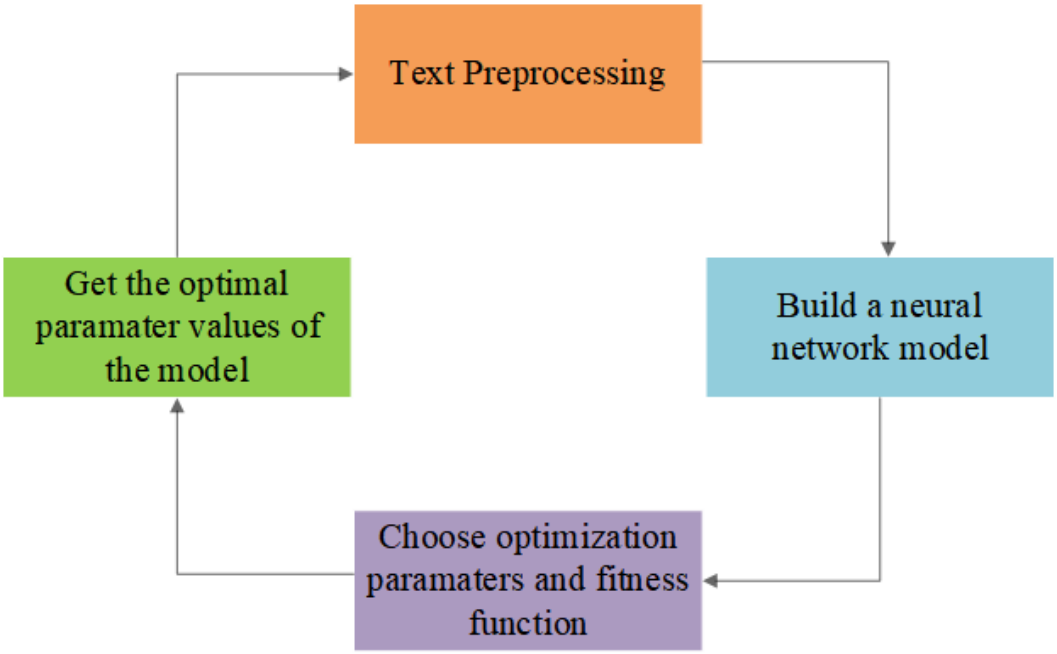
Figure 2. BA network simulation diagram



Text Sentiment Classification Problem

According to the different classification methods, the current text sentiment classification methods can generally be divided into three categories: Sentiment dictionary-based methods, machine learning-based methods, and deep learning-based sentiment classification methods (Mei et al., 2022). The artificial building of a sentiment dictionary is necessary for the sentiment analysis approach based on a sentiment dictionary, which incurs significant labor and time costs (Wang & Yang, 2021). Machine learning-based methods rely on the construction of text features, which are often difficult to extract accurately (Hatzivassiloglou & McKeown, 1997; Pang et al., 2002; Shan et al., 2022). Deep learning-based approaches can efficiently capture context-related semantic data and can automatically extract the right features from the text (Kim, 2014; Sun & Chu, 2020; Yan et al., 2022). This paper chooses the text sentiment classification method based on deep learning as the solution method for this problem. However, due to the difficulty of adjusting hyperparameters in deep learning, it is necessary to apply a parameter optimization algorithm to the sentiment classification model, which is helpful to effectively select hyperparameters.

Figure 3. Sentiment Analysis Process Based on the Parameter Optimization Algorithm



The main process of the sentiment analysis model based on parameter optimization is shown in Figure 3. Text preprocessing operations, such as text cleaning, text segmentation, indexing, and length normalization, are performed. Then a neural network model is established, and the optimization model acts on the parameter tuning of the model. Taking the loss function of the validation set as the fitness function, the optimal values of the hyperparameters, such as batch size and dropout probability in the model are finally obtained.

Text preprocessing takes words as the basic unit, roughly including cleaning, word segmentation, deleting stop words, indexing, and length normalization. Firstly, scan and filter the original data, correct irregular data, and remove useless data. Secondly, use the word segmentation tool jieba to segment the text and then delete the meaningless words in the text. Use the Word2vec model based on Skip-Gram to convert each annotation text into an index list, and each index corresponds to a word in the word vector model. The text preprocessing operation is done by normalizing the index list length to $E\left(L\right)+2\sqrt{D\left(L\right)}$ ($E\left(L\right)$ is the mean of length L and $D\left(L\right)$ is the variance of L).

The embedding layer of the neural network model is represented by a word vector matrix M, whose dimension is $N*D$, where N is the size of the word used and D is the word vector dimension. By converting the index obtained after the text preprocessing into the corresponding word vector, splicing the word vectors of all words in the sentence, and comparing the word vector matrix, the matrix representation of the sentence can be obtained.

The Bi-directional Long Short-Term Memory (BiLSTM) layer is generally used as the main processing unit in text sentiment analysis under deep learning. BiLSTM is a combination of forwarding Long Short-Term Memory (LSTM) and reversing LSTM. BiLSTM can increase classification accuracy by more accurately capturing bidirectional semantic relationships. In BiLSTM, $\overrightarrow{LSTM}$ reads data from left to right, $\overleftarrow{LSTM}$ reads data from right to left, and outputs forward hidden state $\overrightarrow{h_t}$ and reverse hidden state $\overleftarrow{h_t}$, respectively:

$$\overrightarrow{h_t} = \overrightarrow{LSTM}\left(w_t, \overrightarrow{h_{t-1}}, c_{t-1}\right) \tag{6}$$

$$\overleftarrow{h_t} = \overrightarrow{LSTM}\left(w_t, \overleftarrow{h_{t-1}}, c_{t-1}\right) \tag{7}$$

Among them, w_t is the input at time *t*, $\overrightarrow{h_{t-1}}$ and $\overleftarrow{h_{t-1}}$ indicate the concealed layer's condition at a previous time, and c_{t-1} represents the memory storage unit. Finally, $\overrightarrow{h_t}$ concatenates $\overleftarrow{h_t}$ into a long vector that is the input to the next layer:

$$h_t = \overrightarrow{h_t} \oplus \overleftarrow{h_t} \tag{8}$$

The output layer of the neural network model is the sentiment classification layer, which takes the feature information learned by the BiLSTM layer as input to the fully connected layer. This study utilizes the sigmoid activation function in the fully connected layer. The sigmoid classifier maps the output value to the interval between 0 and 1 to obtain the binary representation of emotion. The closer it is to 1, the closer the emotion category is to the positive.

HYBRID LEARNING PARTICLE SWARM OPTIMIZATION WITH FUZZY LOGIC

Mainstream Learning Strategies Based on Scale-free Networks

To solve the problem of poor convergence, a mainstream learning strategy based on the scale-free structure is proposed in this paper. The public is impacted by mainstream views as well as the best people. To simulate this social phenomenon, this paper proposes the mainstream particle $MPbest_i^t$, which is the average value of the individual optimal positions of all particles in the neighborhood of particle *i*, giving full play to the influence of the neighborhood on the particles. The specific process of mainstream learning strategies is as follows:

First, a scale-free network topology is constructed using the BA model: Obtain the Euclidean distance from each particle to the best one, take the nearest m_0 particles as the elite particles to form a fully connected network by elite particles, and then connect the remaining particles to the network with a certain probability.

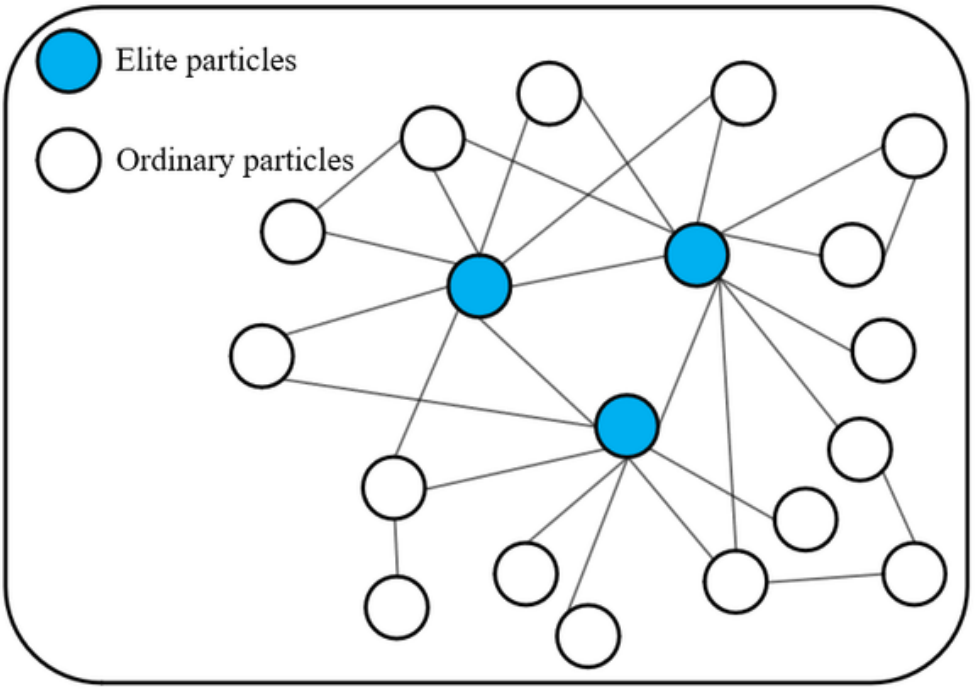
Then, the velocity update is performed using the mainstream particle $MPbest_i^t$ instead of the global optimum in the standard PSO, in which $MPbest_i^t$ is defined as follows:

$$MPbest_i^t = \frac{1}{n}\sum_{j=1}^{n} Pbest_{index_i^j}^t \tag{9}$$

Among them, $index_i^j \left(j = 1,2, \ldots, n\right)$ indicates that the $index_i^j$ particle is in the neighborhood of the i *-th* particle.

Finally, the topology shown in Figure 4 can be obtained. Since the scale-free network is neither as dense as a fully connected network nor as sparse as a ring network, maintaining individual diversity and convergence effectiveness can be balanced via the particle swarm optimization approach.

Figure 4. Scale-free network topology



The main pseudocode of mainstream learning strategies based on scale-free networks is as follows:

```
Strategy1: Mainstream learning strategy based on scale-free
networks
1:          Output: MPbest;
2:          Calculate and sort the distance between particles and
Gbest;
3:          Select the first m_0 elite particles as the initial
nodes to form the network;
4:          Calculate the degree of each node in the network:
Degree=m_0 -1;
5:          For i = m_0 +1 : N
6:           Use the roulette selection algorithm to connect the
particle to the network;
7:           Add 1 to the degree of particle i and connected
particle;
8:          End For
9:          For i = 1 : N
10:           Find the indices of all particles connected to
particle i;
11:           Calculate the average of the historical optimal
positions of these particles as MPbest_i^t ;
12:          End For
```

Stochastic Learning Strategy

A PSO algorithm generally adopts a self-learning strategy, that is, learning from its historical optimal solution. This learning strategy gives the PSO algorithm the advantage of convergence and reliability. However, this strategy leads to the shortcomings of premature convergence and poor performance of the PSO algorithm in dealing with complex problems. To avoid these problems, a random learning strategy is introduced.

Stochastic learning endows particles with the ability to learn from other outstanding individuals in the group, making particle motion more diverse and increasing the diversity of the population. The strategy is described in Equations (10) and (11). In each iteration, for each particle, two different particles are randomly selected from the population, the individual history optimal of the two particles is compared, and the better individual history optimal is selected as the candidate individual optimal solution ($CPbest_i^t$). Then, compare $CPbest_i^t$ with the fitness of the current particle to get the final random individual optimal ($SPbest_i^t$); thus, the particle will use $SPbest_i^t$ to replace it's individual historical optimal to update its speed.

$$CPbest_i^t = argmin\left\{f\left(Pbest_a^t\right),\ f\left(Pbest_b^t\right)\right\} \tag{10}$$

$$SPbest_i^t = argmin\left\{f\left(CPbest_i^t\right), f\left(Pbest_i^t\right)\right\} \tag{11}$$

The pseudocode for the stochastic learning strategy is as follows:

```
Strategy2: Stochastic learning strategy
1:          Output: SPbest_i;
2:          For i = 1:N
3:            Randomly select two particles, particle a, and
particle b;
4:            if f(Pbest_a) < f(Pbest_b) do
5:                CPbest_i = Pbest_a;
6:            else
7:               CPbest_i = Pbest_b;
8:            End if
9:            if f(CPbest_i) < f(Pbest_i) do
10:               SPbest_i = CPbest_i;
11:           else
12:               SPbest_i = Pbest_i;
13:           End if
14:         End For
```

Parameter Self-tuning Based on Fuzzy Logic

Nobile et al. (2015) proposed a self-tuning version of a PSO, known as active particle swarm optimization (PPSO), which uses fuzzy logic to dynamically control parameters, such as inertial weights. Compared with a standard PSO, the convergence and the solution accuracy have been improved by this algorithm. Referring to PPSO, this paper proposes the following self-tuning strategies.

For example, w_i^t, $c_{cog_i^t}$, and $c_{soc_i^t}$ represent the inertia weight, self-learning factor, and social learning factor of the i-th particle during the t-th iteration, respectively. These three parameters are dynamically determined according to fuzzy logic, which mainly depends on two concepts: the distance δ_i^t of the particle to the current global optimal position and the normalized fitness increment factor ϕ_i^t. The distance between particle i and particle j can be calculated by Equation (12):

$$\delta\left(x_i^t, x_j^t\right) = \left\| x_i^t - x_j^t \right\|_2 = \sqrt{\sum_{d=1}^{D}\left(x_i^t\left(d\right) - x_j^t\left(d\right)\right)^2} \tag{12}$$

Among them, $x_i^t\left(d\right)$ and $x_j^t\left(d\right)$ represent the *d*-th dimension of the positions X_i^t and X_j^t of the i*-th* particle and the j-th particle, respectively.

The normalized fitness increment factor ϕ is a variable that measures the improvement of fitness compared with the previous iteration, which fully considers the current position of particle *i* and the position change of the previous iteration. It is defined as follows:

$$\phi_i^t = \frac{min\left\{f\left(x_i^t\right), f_\Delta\right\} - min\left\{f\left(x_i^{t-1}\right), f_\Delta\right\}}{\left|f_\Delta\right|} \cdot \frac{\delta\left(x_i^t, x_i^{t-1}\right)}{\delta_{max}} \tag{13}$$

Among them, δ_{max} represents the distance between the upper and lower bounds of the position and f_Δ represents the worst fitness value of the research problem, which is estimated using the worst fitness value in the iterative process.

The domain of ϕ_i^t is $\left[0,\ \delta_{max}\right]$, and the linguistic value of this variable has the same (Same), near (Near), and far (Far); the following is the membership function of the linguistic value of ϕ_i^t:

$$\text{Same} = \begin{cases} 1, 0 < {}'_i^t \le {}'_1 \\ \dfrac{{}' - {}'_i^t}{{}'_2 - {}'_1}, {}'_1 \le {}'_i^t < {}'_2 \\ 0, {}'_2 \le {}'_i^t \le {}'_{\max} \end{cases} \tag{14}$$

$$\text{Near} = \begin{cases} 0, 0 \le {}'_i^t < {}'_1 \\ \dfrac{{}'_i^t - {}'}{{}' - {}'_1}, {}'_1 \le {}'_i^t < {}'_2 \\ \dfrac{{}' - {}'_i^t}{{}'_3 - {}'_2}, {}'_2 \le {}'_i^t < {}'_3 \\ 0, {}'_3 \le {}'_i^t \le {}'_{\max} \end{cases} \tag{15}$$

$$\mathrm{Far}=\begin{cases}0,0\le {}'^{\mathrm{t}}_{\mathrm{i}}<{}'_{2}\\ \dfrac{{}'^{\mathrm{t}}_{\mathrm{i}}-{}'_{2}}{{}'_{3}-{}'_{2}},{}'_{2}\le {}'^{\mathrm{t}}_{\mathrm{i}}<{}'_{3}\\ 1,{}'_{3}\le {}'^{\mathrm{t}}_{\mathrm{i}}\le {}'_{\max}\end{cases} \tag{16}$$

Among them, δ_1, δ_2, and δ_3 are parameters introduced to describe the fuzzy distance between the particle position and the global optimum. In this paper, refer to the settings in Fuzzy Self-Tuning Particle Swarm Optimization (FSTPSO) (Nobile et al., 2018), that is, $\delta_1 = 0.2 * \delta_{max}$, $\delta_2 = 0.4 * \delta_{max}$, and $\delta_3 = 0.6 * \delta_{max}$.

The domain of the normalized fitness increment factor ϕ_i^t is [-1, 1], and its language values are better (Better), the same (Same), and worse (Worse), and the membership functions are as follows:

$$\mathrm{Better}=\begin{cases}-\phi^{\mathrm{t}}_{\mathrm{i}},-1\le \phi^{\mathrm{t}}_{\mathrm{i}}<0\\ 0,0\le \phi^{\mathrm{t}}_{\mathrm{i}}\le 1\end{cases} \tag{17}$$

$$\mathrm{Same}=1-\left|\phi^{\mathrm{t}}_{\mathrm{i}}\right| \tag{18}$$

$$\mathrm{Worse}=\begin{cases}0,-1\le \phi^{\mathrm{t}}_{\mathrm{i}}<0\\ \phi^{\mathrm{t}}_{\mathrm{i}},0\le \phi^{\mathrm{t}}_{\mathrm{i}}\le 1\end{cases} \tag{19}$$

The above membership functions are all triangular membership functions; after calculating the linguistic value membership degrees of δ_i^t and ϕ_i^t through the above membership functions. Then, according to the fuzzy logic in Table 1, the membership degree of each rule can be obtained by using the cumulative membership method, that is, the membership degree of each language value of w_i^t, $c_{cog_i^t}$, and $c_{soc_i^t}$. After that, it is defuzzified by the weighted average method, the language value is converted into a numerical value, and the calculation method is as follows:

$$output=\frac{\sum_{r=1}^{R}\rho_r z_r}{\sum_{r=1}^{R}\rho_r} \tag{20}$$

Table 1. Fuzzy logic

Output variable	Language value	Definition
w_i^t	Low	ϕ_i^t is Worse, or δ_i^t is Same
	Medium	ϕ_i^t is Same, or δ_i^t is Near
	High	ϕ_i^t is Better, or δ_i^t is Far
$c_{cog_i^t}$	Low	δ_i^t is Far
	Medium	ϕ_i^t is Worse, or δ_i^t is Same, or δ_i^t is Near
	High	ϕ_i^t is Better
$c_{soc_i^t}$	Low	ϕ_i^t is Better, or δ_i^t is Near
	Medium	ϕ_i^t is Same, or δ_i^t is Same
	High	ϕ_i^t is Worse, or δ_i^t is Far

Among them, ρ_r is the membership degree of the input variable to the rule r and z_r the output value of the corresponding rule, as shown in Table 2.

Table 2. Output variables

Output variable	Language value	Number
w_i^t	Low	0.3
	Medium	0.5
	High	1.0
$c_{cog_i^t}$	Low	1.0
	Medium	2.0
	High	3.0
$c_{soc_i^t}$	Low	0.1
	Medium	1.5
	High	3.0

Using the above method, the corresponding values of c_{cog}, c_{soc}, and w can be obtained according to ϕ and δ, and the fuzzy logic can be introduced into the particle swarm optimization algorithm.

The state of the particles can be fully considered, appropriate parameters can be dynamically assigned to each particle, and the development and exploration capabilities can be dynamically adjusted.

The pseudocode of the parameter self-tuning strategy based on fuzzy logic is as follows:

```
Strategy3: Parameter self-tuning strategy based on fuzzy logic
1:        Output: Inertia weight w, self-learning factor c_cog,
social learning factor c_soc;
2:        For i = 1 : N
3:          Calculate the distance δ_i^t from particle i to Gbest;
4:          Calculate the normalized fitness increment factor φ_i^t
for particle i;
5:        End For
6:        For i = 1 : N
7:          Take φ_i^t and δ_i^t as inputs into the membership function
to calculate the membership of φ_i^t and δ_i^t;
8:          Put the membership of φ_i^t and δ_i^t into the fuzzy logic;
9:          Get the language value membership of w_i, c_cog_i^t and c_soc_i^t;
10:         Get the final values of w_i, c_cog_i^t and c_soc_i^t by
defuzzification;
11:       End For
```

Proposed HLPSO-FL Algorithm

To sum up, the velocity and position update equations of HLPSO-FL are as follows:

$$V_i^t = w_i^t \cdot V_i^{t-1} + c_{cog_i^t} \cdot r_1 \cdot \left(SPbest_i^t - X_i^t\right) + c_{soc_i^t} \cdot r_2 \cdot \left(MPbest_i^t - X_i^t\right) \tag{21}$$

$$X_i^t = X_i^t + V_i^t \tag{22}$$

Among them, r_1 and r_2 are two random variables uniformly distributed in (0, 1), $w_i^t, c_{cog_i^t}, c_{soc_i^t}$ are dynamically controlled based on fuzzy logic, $SPbest_i^t$ is determined according to a stochastic learning strategy, and $MPbest_i^t$ is determined by a mainstream learning strategy based on scale-free networks.

After the initialization of the relevant population, the algorithm evaluates the solutions in the population to obtain the fitness value. Then, according to the mainstream learning strategy, the mainstream particles are selected, and the scale-free network topology based on the current population is constructed. At the same time, for each particle, the random learning strategy is used to select the individual particle to be learned from by the population for learning, and the population evolution is completed according to this rule in turn, and the fuzzy rules are used to optimize the algorithm parameters.

The pseudocode of HLPSO-FL algorithm is as follows:

```
Algorithm: Hybrid learning particle swarm optimization with fuzzy
logic
1: Initialization: velocity V, position X;
```

```
2:        Evaluate the fitness of each particle and the global
and historical optima;
3:        While FEs < maxFEs :
4:          Get w_i^t , c_{cog_i^t} , c_{soc_i^t} using Strategy 3;
5:          Get MPbest using Strategy 1;
6:          Get SPbest using Strategy 2;
5:          Replacing the global and historical optima with
MPbest and SPbest ;
6:          Update velocity and location;
7:          Evaluate particle fitness;
8:          Update historical optima and global optima;
9:        End While
```

EXPERIMENTAL RESULTS AND ANALYSIS

To evaluate the HLPSO-FL algorithm's performance, this paper tests the 12 benchmark functions in Table 3 on 10 and 30 dimensions, respectively, and compares them with four well known PSO variants, including Standard PSO (SPSO) (Kennedy & Eberhart, 1995), Comprehensive learning PSO (CLPSO) (Lynn & Suganthan, 2015), Social Learning PSO (SLPSO) (Cheng & Jin, 2015), and Biogeography-based learning PSO (BLPSO) (Chen et al., 2020). To make sure the algorithm comparison is fair, each algorithm is to be executed independently 30 times, the population size is 30, the maximum number of fitness evaluation times is D*10000, and the optimal value, average value, and standard deviation of the 30 running results are recorded. The priority order of mean value (Mean), standard deviation (Std), and optimal value (Min) are used to sort from small to large. Experimental data of algorithm using bold font to obtain optimal results. Tables 5 and 6 display the results of the final experiment. Table 4 clearly describes the parameter settings in each algorithm.

Table 3. Benchmark functions

Function	Variable scope	The optimal value
f_1 : Sphere	[–100, 100]	0
f_2 : Schwefel 2.22	[–10, 10]	0
f_3 : Schwefel 1.2	[–100, 100]	0
f_4 : Schwefel 2.21	[–100, 100]	0
f_5 : Rosenbrock	[–30, 30]	0
f_6 : Quartic	[–1.28, 1.28]	0
f_7 : Rastrigin	[–5.12, 5.12]	0
f_8 : Griewank	[–600, 600]	0

Table 3 continued on next page

Table 3 continued

Function	Variable scope	The optimal value
f_9 : Ackley	[–32, 32]	0
f_{10} : Weierstrass	[–0.5, 0.5]	0
f_{11} : Xin-She Yang N.2	$\left[-2\grave{A}, 2\grave{A}\right]$	0
f_{12} : Alpine 1	[–10, 10]	0

Table 4. Parameter settings of the comparison algorithm

Algorithm	Parameter settings
SPSO	$w = 0.9 \sim 0.4, c_1 = c_2 = 2$
CLPSO	$w = 0.9 \sim 0.4, c = 1.49445, gapm = 5$
SLPSO	$m = popSize + D / 10, \alpha = 0.5, \beta = 0.01$
BLPSO	$w = 0.9 \sim 0.2, c = 1.49445, I = E = 1, gapm = 5$
HLPSO-FL	$m_0 = \left[0.3 * popSize\right], \delta_1 = 0.2 * \delta_{max}$, $\delta_2 = 0.4 * \delta_{max}, and\, \delta_3 = 0.6 * \delta_{max}$

Table 5. Comparative experimental results (D = 10)

		SPSO	CLPSO	SLPSO	BLPSO	HLPSO-FL
f_1	Mean	2.07E–80	8.05E–29	**4.46E–288**	6.38E–60	2.04E–238
	Std.	7.77E–80	1.60E–28	**0.00E+00**	1.63E–59	0.00E+00
	Min	1.39E–88	1.38E–30	**3.78E–296**	9.30E–65	5.92E–248
	Rank	3	5	**1**	4	2
f_2	Mean	1.04E–46	2.01E–17	**5.57E–148**	3.66E–38	2.30E–119
	Std.	2.76E–46	1.44E–17	**1.74E–147**	1.71E–37	7.89E–119
	Min	3.21E–49	2.75E–18	**1.03E–152**	5.41E–41	5.61E–126
	Rank	3	5	**1**	4	2

Table 5 continued on next page

Table 5 continued

		SPSO	CLPSO	SLPSO	BLPSO	HLPSO-FL
f_3	Mean	4.11E–24	8.34E–02	1.29E+00	1.78E–07	**5.36E–45**
	Std.	1.93E–23	7.49E–02	4.42E+00	6.42E–07	**2.21E–44**
	Min	1.86E–29	1.37E–02	3.70E–05	6.71E–12	**8.22E–53**
	Rank	2	4	5	3	**1**
f_4	Mean	4.57E–20	1.79E–01	3.38E–02	2.68E–01	**7.81E–77**
	Std.	1.93E–19	6.16E–02	1.25E–01	4.23E–01	**2.44E–76**
	Min	1.50E–23	6.06E–02	2.00E–66	2.72E–03	**1.52E–82**
	Rank	2	4	3	5	**1**
f_5	Mean	3.11E+03	2.04E+00	8.70E+01	4.13E+00	**2.56E–01**
	Std.	1.61E+04	1.71E+00	1.53E+02	4.38E+00	**6.94E–01**
	Min	4.02E–02	3.96E–01	4.35E+00	9.18E–02	**5.68E–02**
	Rank	5	2	4	3	**1**
f_6	Mean	1.23E–03	1.35E–03	6.59E–03	2.06E–03	**3.13E–04**
	Std.	6.17E–04	5.88E–04	3.97E–03	3.71E–03	**1.22E–04**
	Min	3.05E–04	3.03E–04	1.05E–03	1.61E–04	**9.94E–05**
	Rank	2	3	5	4	**1**
f_7	Mean	4.26E+00	**0.00E+00**	7.96E+00	1.89E+00	2.90E+00
	Std.	2.65E+00	**0.00E+00**	3.26E+00	1.21E+00	3.16E+00
	Min	0.00E+00	**0.00E+00**	1.99E+00	0.00E+00	0.00E+00
	Rank	4	**1**	5	2	3
f_8	Mean	6.82E–02	**1.52E–03**	2.84E–02	3.72E–02	2.77E–02
	Std.	3.31E–02	**3.19E–03**	1.84E–02	3.70E–02	3.61E–02
	Min	7.40E–03	**1.21E–07**	0.00E+00	0.00E+00	0.00E+00
	Rank	5	**1**	3	4	2
f_9	Mean	1.98E–03	1.93E–14	3.85E–02	1.36E+00	**2.93E–15**
	Std.	1.07E–02	1.36E–14	2.07E–01	9.32E–01	**1.63E–15**
	Min	4.00E–15	4.00E–15	4.00E–15	4.00E–15	**4.44E–16**
	Rank	3	2	4	5	**1**
f_{10}	Mean	5.19E–02	**0.00E+00**	5.93E–02	3.76E–02	**0.00E+00**
	Std.	2.69E–01	**0.00E+00**	2.69E–01	1.07E–01	**0.00E+00**
	Min	0.00E+00	**0.00E+00**	0.00E+00	0.00E+00	**0.00E+00**
	Rank	4	**1**	5	3	**1**
f_{11}	Mean	1.21E–03	**5.66E–04**	1.30E–03	5.86E–04	6.73E–04
	Std.	3.98E–04	**6.79E–09**	3.30E–04	3.65E–05	2.32E–04
	Min	5.66E–04	**5.66E–04**	6.37E–04	5.66E–04	5.66E–04
	Rank	4	**1**	5	2	3

Table 5 continued on next page

Table 5 continued

		SPSO	CLPSO	SLPSO	BLPSO	HLPSO-FL
f_{12}	Mean	3.23E–15	5.05E–05	**4.72E–16**	9.46E–16	1.08E–10
	Std.	2.78E–15	4.24E–05	**9.86E–16**	1.14E–15	5.82E–10
	Min	3.16E–75	5.75E–06	**0.00E+00**	5.92E–80	1.49E–62
	Rank	3	5	**1**	2	4
Total rank		40	34	42	41	22
Final rank		3	2	5	4	1

In Table 5, HLPSO-FL has good development ability in functions f_3, f_4, f_5, f_6, f_9, and f_{10}, and performs the best among these algorithms. The convergence of functions f_1 and f_2 is slightly worse than SLPSO but better than the other three optimization algorithms. CLPSO, BLPSO, and HLPSO-FL can all converge to the optimal value on f_7, but the stability of HLPSO-FL is slightly worse than the former two. The main reason is that the learning strategies of CLPSO and BLPSO are for the entire particle swarm. However, part of the learning strategy of HLPSO-FL is based on the scale-free network topology and uses a random learning strategy, so it is more random and slightly less stable. Although HLPSO-FL performed poorly on f_7, f_{11}, and f_{12}, ranking 3rd, 3rd, and 4th, respectively, the overall performance of HLPSO-FL showed significant superiority.

Table 6. Comparative experimental results (D = 30)

		SPSO	CLPSO	SLPSO	BLPSO	HLPSO-FL
f_1	Mean	9.34E–59	5.84E–28	4.56E–07	1.11E–57	**3.53E–264**
	Std.	2.68E–58	4.46E–28	1.31E–06	2.42E–57	**0.00E+00**
	Min	6.14E–63	1.00E–28	3.54E–28	1.52E–65	**3.45E–282**
	Rank	2	4	5	3	**1**
f_2	Mean	2.67E+00	6.58E–17	2.20E–02	2.86E–36	**8.84E–132**
	Std.	5.12E+00	3.05E–17	1.10E–01	1.17E–35	**4.76E–131**
	Min	3.87E–41	2.39E–17	8.87E–09	2.78E–42	**5.88E–155**
	Rank	5	3	4	2	**1**
f_3	Mean	7.67E+03	5.23E+02	4.57E+03	2.11E+00	**4.33E–11**
	Std.	5.69E+03	1.56E+02	3.23E+03	9.47E+00	**6.67E–11**
	Min	2.17E–02	2.50E+02	9.21E+02	3.34E–02	**8.66E–13**
	Rank	5	3	4	2	**1**
f_4	Mean	7.93E–01	2.24E+00	2.73E+01	1.11E+01	**1.29E–10**
	Std.	5.88E–01	3.89E–01	5.73E+00	4.11E+00	**6.68E–10**
	Min	1.42E–01	1.59E+00	1.79E+01	3.28E+00	**7.11E–17**
	Rank	2	3	5	4	**1**

Table 6 continued on next page

Table 6 continued

		SPSO	CLPSO	SLPSO	BLPSO	HLPSO-FL
f_5	Mean	9.24E+03	**8.31E–01**	2.56E+02	3.85E+01	6.45E+00
	Std.	2.69E+04	**5.35E–01**	5.39E+02	4.92E+01	4.19E+00
	Min	2.95E–01	**1.52E–01**	1.15E+01	4.35E–01	2.64E–02
	Rank	5	**1**	4	3	2
f_6	Mean	9.07E–03	4.19E–03	2.26E–01	2.13E–02	**1.65E–03**
	Std.	2.69E–03	1.24E–03	9.64E–01	1.58E–02	**7.52E–04**
	Min	3.59E–03	2.45E–03	1.24E–02	2.04E–03	**6.79E–04**
	Rank	3	2	5	4	**1**
f_7	Mean	6.83E+01	**3.32E–02**	6.14E+01	1.38E+01	1.47E+01
	Std.	2.31E+01	**1.79E–01**	1.89E+01	4.36E+00	5.83E+00
	Min	2.40E+01	**0.00E+00**	3.28E+01	6.96E+00	7.96E+00
	Rank	5	**1**	4	2	3
f_8	Mean	3.05E+00	**2.76E–10**	2.96E–02	1.35E–01	7.87E–03
	Std.	1.63E+01	**1.16E–09**	4.90E–02	1.69E–01	1.23E–02
	Min	0.00E+00	**5.44E–15**	0.00E+00	0.00E+00	0.00E+00
	Rank	5	**1**	3	4	2
f_9	Mean	9.20E–01	**3.44E–14**	1.58E+00	3.79E+00	6.01E–01
	Std.	9.52E–01	**8.19E–15**	1.16E+00	9.53E–01	7.08E–01
	Min	7.55E–15	**2.53E–14**	1.21E–10	2.22E+00	4.00E–15
	Rank	3	**1**	4	5	2
f_{10}	Mean	3.06E+00	**6.16E–15**	3.72E+00	1.97E+00	5.30E–01
	Std.	2.17E+00	**2.42E–15**	1.61E+00	8.98E–01	6.97E–01
	Min	1.13E–01	**0.00E+00**	1.15E+00	7.75E–01	1.13E–04
	Rank	4	**1**	5	3	2
f_{11}	Mean	1.57E–11	5.15E–12	9.42E–12	**5.12E–12**	1.15E–09
	Std.	3.96E–12	8.61E–13	1.08E–12	**8.54E–13**	2.27E–09
	Min	9.32E–12	3.65E–12	6.20E–12	**3.66E–12**	3.51E–12
	Rank	4	2	3	**1**	5
f_{12}	Mean	7.40E–01	5.18E–04	4.62E–05	5.95E–15	**8.51E–16**
	Std.	1.65E+00	2.88E–04	2.28E–04	9.71E–15	**7.39E–16**
	Min	6.88E–15	1.74E–04	1.09E–11	1.28E–15	**1.44E–21**
	Rank	5	4	3	2	**1**
Total rank		48	26	49	35	22
Final rank		5	2	4	3	1

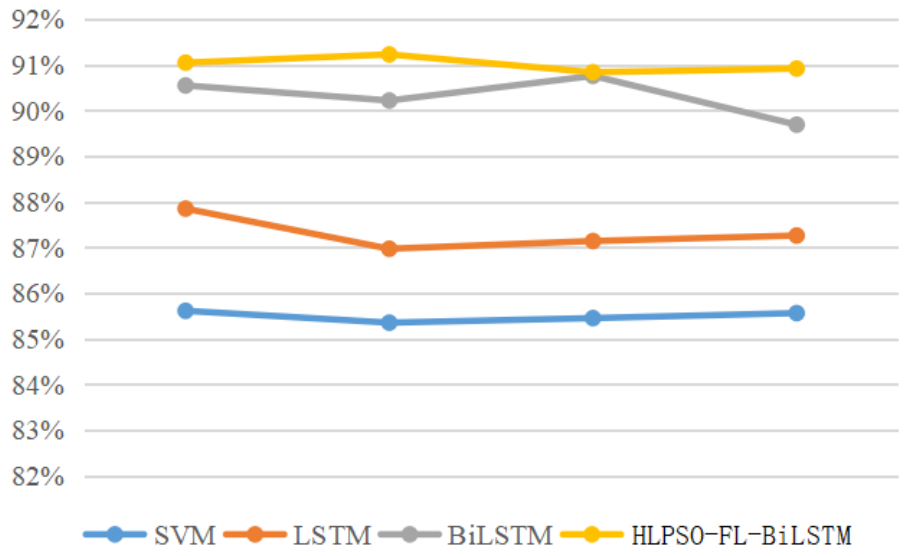
By analyzing Table 6, it can be seen that on 30-dimensional data, HLPSO-FL has excellent development ability in functions f_1, f_2, f_3, f_4, f_6, and f_{12}, and the convergence and stability are better than the others. Although the results of HLPSO-FL on functions f_7 and f_{11} are not impressive, ranking 3rd and 5th, respectively, it ranks 2nd on functions f_5, f_8, f_9, and f_{10}, and ranks first overall, with good competitiveness.

In conclusion, although HLPSO-FL performs slightly worse in some cases, on the whole, after comprehensive sorting, the stability and convergence accuracy of the HLPSO-FL algorithm in solving single-modal and multi-modal problems show excellent competitiveness and superiority.

To verify the utility of the proposed HLPSO-FL in the text sentiment analysis problem, experiments are next conducted on real problems. This experiment adopts the deep learning framework of Python 3.8 + Keras 2.3.1, and the data set used is the hotel review data set collected by Mr. Tan Songbo (https://github.com/lunarwhite/Chinese-corpus-sentiment-analysis). The word vector model uses a pretrained word vector model developed by researchers from the Chinese Information Processing Institute of Beijing Normal University and the Database and Intelligent Information Retrieval Laboratory of the Renmin University of China, where the dimension of the word vector is 300 dimensions (Li et al., 2018). Each layer of the neural network has a Dropout layer added to it, and an early stop technique is also used to prevent overfitting. When the loss value of the validation set of three iterations is not improved, the training is stopped. At the same time, the automatic decay factor of the learning rate is set to 0.1, and the training is stopped when the learning rate decays to the minimum value of 10^{-5}.

This experiment uses HLPSO-FL to optimize the three hyperparameters of batch size, dropout probability, and the number of hidden layer neurons in the neural network. The optimization ranges are 30~300, 0.5~0.6, and 10~100, respectively. Finally, the optimal values of hyperparameters are 128, 0.2, and 32. To verify the feasibility of the model proposed in this paper, three comparison models of Support Vector Machine (SVM), LSTM, and BiLSTM are selected. The average of the trial results is calculated after each model has been trained 10 times. The classification results of different models are as follows:

Figure 5. Classification results of different models



In Figure 5, HLPSO-FL-BiLSTM is compared with the traditional machine learning classification model SVM, the accuracy rates of LSTM, BiLSTM, and HLPSO-FL-BiLSTM are 2.24%, 4.94%, and 5.44% higher than that of SVM, respectively. The side reflects that the deep learning model has obvious advantages. Compared with the LSTM model, the BiLSTM model has a 3.25% increase in the F1 value (*f1-score*) and a 3.62% increase in the accuracy rate. This is because BiLSTM can capture bidirectional semantic information. Compared with the BiLSTM model, HLPSO-FL-BiLSTM improves the F1 value by 1.01% and the recall rate by 1.23%, indicating that proper hyperparameters improve classification accuracy and performance.

CONCLUSION

To solve the problem of premature convergence and diversity loss of a particle swarm optimization algorithm, a hybrid learning particle swarm optimization method based on fuzzy logic is proposed, which mainly includes three improvement strategies. First of all, based on the scale-free network topology, we propose a mainstream learning strategy, which ensures the convergence of the algorithm, while reducing the speed of information transmission and helps to avoid local optimization. Secondly, the use of random learning strategies instead of self-learning strategies helps to enrich the diversity and avoid premature convergence. In addition, according to the state of each particle, fuzzy logic is used to dynamically control the inertia weight, individual learning factor, and other parameters of each particle, and dynamically adjust the exploration and development capability.

Experiments are employed to confirm the algorithm's efficacy, and it is used to optimize the hyperparameters in the sentiment analysis model. The results show that the algorithm can effectively select suitable hyperparameters. However, the algorithms and models proposed in this paper still have shortcomings, such as insufficient convergence proof and corresponding theoretical analysis of the algorithm, and incomplete classification of sentiment analysis models. Therefore, our next work will focus on summarizing the theoretical model and classification of the emotion analysis problem, so that this problem can be better solved.

ACKNOWLEDGMENT

This work was supported by the Graduate Innovation Foundation of Jiangxi University of Science and Technology (Grant No. XY2022-S040).

COMPETING INTERESTS

The authors declare that they have no conflicts of interest in this work.

REFERENCES

Barabási, A. L., & Albert, R. (1999). Emergence of scaling in random networks. *Science*, *286*(5439), 509–512. doi:10.1126/science.286.5439.509 PMID:10521342

Cao, Y., Zhang, H., Li, W., Zhou, M., Zhang, Y., & Chaovalitwongse, W. A. (2018). Comprehensive learning particle swarm optimization algorithm with local search for multimodal functions. *IEEE Transactions on Evolutionary Computation*, *23*(4), 718–731. doi:10.1109/TEVC.2018.2885075

Chen, X., Li, K., Xu, B., & Yang, Z. (2020). Biogeography-based learning particle swarm optimization for combined heat and power economic dispatch problem. *Knowledge-Based Systems*, *208*, 106463. doi:10.1016/j.knosys.2020.106463

Cheng, R., & Jin, Y. (2015). A social learning particle swarm optimization algorithm for scalable optimization. *Information Sciences*, *291*, 43–60. doi:10.1016/j.ins.2014.08.039

Ding, H., & Gu, X. (2020). Improved particle swarm optimization algorithm based novel encoding and decoding schemes for flexible job shop scheduling problem. *Computers & Operations Research*, *121*, 104951. doi:10.1016/j.cor.2020.104951

Gupta, I. K., Choubey, A., & Choubey, S. (2017). Particle swarm optimization with selective multiple inertia weights. In *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT*) (pp. 1–6). IEEE. doi:10.1109/ICCCNT.2017.8204132

Hatzivassiloglou, V., & McKeown, K. (1997). Predicting the semantic orientation of adjectives. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics* (pp. 174–181). Academic Press.

He, Z., Liu, T., & Liu, H. (2022). Improved particle swarm optimization algorithms for aerodynamic shape optimization of high-speed train. *Advances in Engineering Software*, *173*, 103242. doi:10.1016/j.advengsoft.2022.103242

Hu, M., Wu, T., & Weir, J. D. (2012). An adaptive particle swarm optimization with multiple adaptive methods. *IEEE Transactions on Evolutionary Computation*, *17*(5), 705–720. doi:10.1109/TEVC.2012.2232931

Kennedy, J., & Eberhart, R. (1995, November). Particle swarm optimization. In *Proceedings of ICNN'95-International Conference on Neural Networks* (Vol. 4, pp. 1942–1948). IEEE. doi:10.1109/ICNN.1995.488968

Kim, Y. (2014). Convolutional neural networks for sentence classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 1746–1751. doi:10.3115/v1/D14-1181

Li, S., Chi, X., & Yu, B. (2022). An improved particle swarm optimization algorithm for the reliability redundancy allocation problem with global reliability. *Reliability Engineering & System Safety*, *225*, 108604. doi:10.1016/j.ress.2022.108604

Li, S., Zhao, Z., Hu, R., Li, W., Liu, T., & Du, X. (2018). Analogical reasoning on Chinese morphological and semantic relations. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (*Vol. 2*: Short Papers),* 138–143. doi:10.18653/v1/P18-2023

Liang, J. J., & Suganthan, P. N. (2005, June). Dynamic multi-swarm particle swarm optimizer. In *Proceedings 2005 IEEE Swarm Intelligence Symposium* (pp. 124–129). IEEE. doi:10.1109/SIS.2005.1501611

Lin, A., Sun, W., Yu, H., Wu, G., & Tang, H. (2019). Global genetic learning particle swarm optimization with diversity enhancement by ring topology. *Swarm and Evolutionary Computation*, *44*, 571–583. doi:10.1016/j.swevo.2018.07.002

Liu, H., Zhang, X. W., & Tu, L. P. (2020). A modified particle swarm optimization using adaptive strategy. *Expert Systems with Applications*, *152*, 113353. doi:10.1016/j.eswa.2020.113353

Liu, Z., Wang, J., Zhang, C., Chu, H., Ding, G., & Zhang, L. (2021). A hybrid genetic-particle swarm algorithm based on multilevel neighbourhood structure for flexible job shop scheduling problem. *Computers & Operations Research*, *135*, 105431. doi:10.1016/j.cor.2021.105431

Lynn, N., & Suganthan, P. N. (2015). Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation. *Swarm and Evolutionary Computation*, *24*, 11–24. doi:10.1016/j.swevo.2015.05.002

Mei, J.-P., Zhen, Y., Zhou, Q., & Yan, R. (2022). TaskDrop: A competitive baseline for continual learning of sentiment classification. *Neural Networks*, *155*, 551–560. doi:10.1016/j.neunet.2022.08.033 PMID:36191451

Nobile, M. S., Cazzaniga, P., Besozzi, D., Colombo, R., Mauri, G., & Pasi, G. (2018). Fuzzy self-tuning PSO: A settings-free algorithm for global optimization. *Swarm and Evolutionary Computation*, *39*, 70–85. doi:10.1016/j.swevo.2017.09.001

Nobile, M. S., Pasi, G., Cazzaniga, P., Besozzi, D., Colombo, R., & Mauri, G. (2015). Proactive particles in swarm optimization: A self-tuning algorithm based on fuzzy logic. *2015 IEEE International Conference on Fuzzy Systems*, 1–8. doi:10.1109/FUZZ-IEEE.2015.7337957

Pang, B., Lee, L., & Vaithyanathan, S. (2002). *Thumbs up? Sentiment classification using machine learning techniques*. arXiv preprint cs/0205070.

Shan, Y., Che, C., Wei, X., Wang, X., Zhu, Y., & Jin, B. (2022). Bi-graph attention network for aspect category sentiment classification. *Knowledge-Based Systems*, *258*, 109972. doi:10.1016/j.knosys.2022.109972

Singh, G., & Singh, A. (2023). Extension of particle swarm optimization algorithm for solving two-level time minimization transportation problem. *Mathematics and Computers in Simulation*, *204*, 727–742. doi:10.1016/j.matcom.2022.09.013

Sun, F., & Chu, N. (2020). Text sentiment analysis based on CNN-BiLSTM-attention model. *2020 International Conference on Robots & Intelligent System*, 749–752. doi:10.1109/ICRIS52159.2020.00186

TanweerM. R.SureshS.SundararajanN. (2016). Dynamic mentoring and self-regulation based particle swarm optimization algorithm for solving complex real-world optimization problems. Information Sciences, 326, 1–24. 10.1016/j.ins.2015.07.035

VandenBerghF.EngelbrechtA. P. (2004). A cooperative approach to particle swarm optimization. IEEE Transactions on Evolutionary Computation, 8(3), 225–239. 10.1109/TEVC.2004.826069

Wang, T., & Yang, W. (2021). Review of text sentiment analysis methods. *Computer Engineering and Applications*, *57*(12), 11–24.

Wang, S., Li, Y., & Yang, H. (2019). Self-adaptive mutation differential evolution algorithm based on particle swarm optimization. *Applied Soft Computing*, *81*, 105496. doi:10.1016/j.asoc.2019.105496

Xia, X., Gui, L., & Zhan, Z.-H. (2018). A multi-swarm particle swarm optimization algorithm based on dynamical topology and purposeful detecting. *Applied Soft Computing*, *67*, 126–140. doi:10.1016/j.asoc.2018.02.042

Xiaodong, L. (2012). Cooperatively coevolving particle swarms for large scale optimization. *IEEE Transactions on Evolutionary Computation*, *16*(2), 210–224. doi:10.1109/TEVC.2011.2112662

Xu, G., Cui, Q., Shi, X., Ge, H., Zhan, Z.-H., Lee, H. P., Liang, Y., Tai, R., & Wu, C. (2019). Particle swarm optimization based on dimensional learning strategy. *Swarm and Evolutionary Computation*, *45*, 33–51. doi:10.1016/j.swevo.2018.12.009

Yan, C., Liu, J., Liu, W., & Liu, X. (2022). Research on public opinion sentiment classification based on attention parallel dual-channel deep learning hybrid model. *Engineering Applications of Artificial Intelligence*, *116*, 105448. doi:10.1016/j.engappai.2022.105448

Yu, H., Tan, Y., Zeng, J., Sun, C., & Jin, Y. (2018). Surrogate-assisted hierarchical particle swarm optimization. *Information Sciences*, *454*, 59–72. doi:10.1016/j.ins.2018.04.062

Zhang, X., Liu, H., Zhang, T., Wang, Q., Wang, Y., & Tu, L. (2019). Terminal crossover and steering-based particle swarm optimization algorithm with disturbance. *Applied Soft Computing*, *85*(105841), 21. doi:10.1016/j.asoc.2019.105841

Jiyuan Wang is working in the School of Information Engineering, Jiangxi University of Science and Technology, Ganzhou, China. He received his master's degree in School of Civil and Surveying & Mapping Engineering from the Jiangxi University of Science and Technology, Ganzhou, China in 2014. His current research interests include artificial intelligence, machine learning and visual computing.

Kaiyue Wang obtained her bachelor's degree in computer science and technology from Jiangxi University of Science and Technology. Her main research direction is computational intelligence.

Xiangfang Yan is studying for a postgraduate degree in computer application technology at Jiangxi University of Science and Technology. His main research interests are computational intelligence and swarm intelligence.

Chanjuan Wang received a postgraduate degree from Jiangxi University of Technology. His main research interests are computer vision, video analysis and machine learning.