Chinese Word Segmentation based on the Improved Particle Swarm Optimization Neural Networks

Jia He
Computational Intelligence Laboratory
School of Computer Science and Engineering, UESTC
Chengdu, China
Department of Computer
Chengdu University of Information Technology
hejia@cuit.edu.cn

Lin Chen
Department of Computer
Chengdu University of Information Technology
Chengdu, China
suelincl@126.com

***Abstract*—The Chinese word segmentation based on the improved Particle Swarm Optimization (PSO) neural networks is discussed in this paper. Firstly, a solution is obtained by searching globally using FPSO (Fuzzy cluster Particle Swarm Optimization) algorithm, which has strong parallel searching ability, encoding real number, and optimizing the training weights, thresholds, and structure of neural networks. Then based on the optimization results obtained from FPSO algorithm, the optimization solution is continuously searched by the following BP algorithm, which has strong local searching ability, until it is discovered finally. Simulation results show that the method proposed in this paper greatly increases both the efficiency and the accuracy of Chinese word segmentation.**

***Keywords*—Particle Swarm Optimization(PSO), Fuzzy cluster Particle Swarm Optimization(FPSO), Chinese word segmentation, BP neural networks**

I. INTRODUCTION

Chinese word segmentation is one of the most difficult research topics in natural language processing. It is the base of Chinese information processing. The smallest components of Chinese language are meaningful words instead of characters[1]. Unlike English language, there is no "blank space" between two Chinese words. How to tell meaningful Chinese words from sentences, which are serially stored in computer memory as Chinese character strings, is called Chinese word segmentation. In the process of Chinese word segmentation by computer, denote an input Chinese character string as $C_1C_2C_3\cdots C_n$ and the output as the sequential Chinese words $W_1W_2W_3\cdots W_m$, where W_i is a Chinese word with either only one or more Chinese characters[4].

Currently, Chinese word segmentation is usually dealt with neural networks especially BP neural networks that is widely used[5][12][13]. However, BP neural networks use gradient descent method and easily get into local minimum. That is, as the number of training samples increased, the input and output relationship becomes complex. The convergence speed of networks becomes slower[11]. In order to overcome this shortage, researchers proposed many methods, such as methods to improve the error function and methods to improve the excitation function etc., to improve training speed of networks, but it still can not avoid local convergence. Convergence speed and accuracy of word segmentation can be improved by the neural networks trained by genetic algorithm at the same time[12]. But the training time will be increased because of the complex operation of genetic algorithm. The accuracy will be decreased by the redundant connections of neural networks.

In our research, Chinese word segmentation based on the improved FPSO neural networks is implemented as follows: Firstly, a solution is obtained by searching globally using FPSO (Fuzzy cluster Particle Swarm Optimization) algorithm, which has strong parallel searching ability, encoding real number, and optimizing the training weights, thresholds, and structure of neural networks. Then based on the optimization results obtained from FPSO algorithm, the optimization solution is continuously searched by following BP algorithm, which has strong local searching ability, until it is discovered finally[3]. Simulation results show that the method proposed in this paper greatly increases both of the efficiency and accuracy of Chinese word segmentation.

II. INTRODUCTION OF FPSO ALGORITHM

PSO optimizing BP neural networks focuses on how the particle swarm can effectively search the optimization solution in the solution space[2]. However, along the increasing of iteration times and swarm types, the basic PSO algorithm is time-consuming and requires a large number of iteration times when calculating values with high accuracy although it is able to converge to the global minimum[6]. Further more, the information between particles may be too concentrated when evolving to a certain number of generations. Thus, most particles gather around few points so that both of the prematurity and stagnation may appear[7]. The basic PSO optimizing BP neural networks is impossible if particles are all premature and stagnation[9]. The difference between FPSO optimizing neural networks and PSO optimizing BP neural networks depends on the time before optimizing the location and speed of PSO. Firstly, the particle swarm is divided into subpopulations by using FCM algorithm and the optimization location of each subpopulation is calculated[8]. Then the particle in the particle swarm updates the values of its speed and location based on its personal best (Pbest) and the optimization location of every subpopulation. Therefore, the performance of FPSO optimizing BP neural networks is better than the PSO optimizing BP neural networks[3].

In studying algorithms based on FPSO optimization, let $x_i=(x_{i1}, x_{i2}, \cdots, x_{id})$ be a group values of parameters, each

dimension of a vector be the weights or threshold, and d be the number of all weights and thresholds in a neural networks. The fitness function of every particle is shown as follows[9]:

$$I_i = \sum_j \left(Y_{i,j} - y_{i,j}\right)^2 \quad (1)$$

$$I_{popIndex} = \frac{1}{n}\sum_{i=1}^{n} I_i \quad (2)$$

Here, n is the number of samples, $Y_{i,j}$ is the jth ideal output value of the ith sample, $Y_{i,j}$ is the actual output value of the ith sample, and popIndex=1, ···, popSize, popSize is the size of the particle types(ie.: the number of particles).

III. DESIGNAND IMPLEMENTATION OF FPSO ALGORITHM OPTIMIZING BP NEURAL NETWORKS

The steps of FPSO optimizing BP networks are shown as follows:

1) Initialize the BP networks, and assign the number of neurons in the input layer, hidden layer, and output layer.

2) Initialize the particle swarm and the speed of every particle:

a) The location of particle, the dimension number of the speed vector (dimSize is the size of the particle types, i.e. the number of particles)

dimSize = the number of connection weights from input layer to hidden layer + the number of connection weights from hidden layer to output layer + the threshold number in hidden layer + the threshold number in output layer

b) While initializing the particle swarm and the speed of every particle, the 1st and 2nd matrix x, all dimensions of the first dimSize columns representing the particle locations, all dimensions of the last dimSize columns representing the particle speeds, and the last dimension representing the fitness of the particle, are initialized firstly.

c) Initialize the Pbest and Gbest of every particle. Here, Pbest is the particle's personal best, and Gbest is the particle's global best.

3) Calculate the fitness of every particle:

a) A particle is input first. For each sample, an output value of the networks can be calculated by using the forwarding method of BP neural networks. Then the error is calculated by formula(1). By following the same process, errors of all samples are calculated. Next, the mean square diviations (i.e: the fitness of a particle) of all samples are calculated by formula(2).

b) Return to step a), and input all other particles until the fitness values of all particles are calculated.

I) In order to calculate conveniently, the elements from column 1 to column IN * HN in the initialized particle matrix are assigned to the weight matrix from input layer to hidden layer. Respectively, elements from column IN * HN+1 to column IN* HN + HN * ON are assigned to the weight matrix from hidden layer to output layer, elements from column IN * HN + HN * ON + 1 to column IN* HN + HN * ON + HN are assigned to thresholds of the hidden layer, and elements from column IN* HN + HN * ON + HN + 1 to column IN * HN + HN * ON + HN + ON are assigned to thresholds of output layer. Here, IN is the number of neurons in the input layer, HN is the number of neurons in the hidden layer, and ON is the number of neurons in the output layer[2].

II) When calculating using the forwarding method of BP neural networks, the Sigmoid function is used in the hidden layer, while the linear Pureline function is used in the output layer. The curve function is adopted in the last layer of a network in BP network models, the output is limited in a very small domain. If the linear function is adopted, the output can be a random value. Thus, the linear function is adopted in the last layer when BP networks are designed in MATLAB, i.e.: f(x)=x.

4) Compare the fitness values, and ensure the Pbest and Gbest of every particle:

If Present < Pbest and Pbest = Present, Pbest= x_i; otherwise, Pbest is not changed;

If Present < Gbest and Gbest=Present, Gbest= x_i; otherwise, Gbest is not changed.

Here, Present is the fitness of the current particle, Pbest is the particle's personal best, and Gbest is the particle's global best.

5) Update the location and speed of every particle:

Let G={x_1, x_2, ···, x_n} represent a data collection composed of n particals x_i (i=1, 2, ···, n), which is divided into K cluster C_i (i=1, 2, ···, K), and sp_i(i=1, 2, ···, K) be the optimization location searched by particles in every cluster C_i until current time. The particles update their speeds and locations by the formula shown as follows[3]:

$$v_i(t+1) = v_i(t) + c_0 r_0(t)\left(pbest_i(t) - x_i(t)\right) + \sum_{j=1}^{K} c_j r_j \left(sp_i(t) - x_i(t)\right) \quad (3)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (4)$$

Here, the learning factor c_j (j=0, 1, ···, K) is a constant, and r_j (j=0, 1, ···, K) is a random number in the domain of [0,1]. The speed and location of particles are updated by formula (3) and (4). Meanwhile, it is considered whether both the updated speed and location are within the limited domains.

6) Calculate the error caused by the algorithm using the on-line performance criterion or the off-line performance criterion. The on-line performance criterion are utilized to evaluate the performance of networks, and the criterion are as follows: If fun(Gbesti)<eg, the algorithm is convergent;

otherwise, iterate continuously until fun(Gbesti) reaches the assigned accuracy. Here, eg is the assigned accuracy for algorithm, and fun(Gbesti) is the fitness of the global best of the ith iteration.

7) Compare if the number reaches the largest iteration number or satisfies the requirements of Step 6). If satisfying the assigned accuracy, the algorithm is convergent. Thus, the weight and threshold of each dimension in the global best Gbest obtained from the last iteration is what we desire; otherwise, return to Step 2), the algorithm is iterated continuously. The BP algorithm presented in this paper, which is based on FPSO optimization algorithm.

From the discussion above, we know that these two algorithms are similar on two aspects:

1) The weights and thresholds of BP network are assigned as the value of each element in vectors of particles.

2) While the fitness of a particle is calculated, the forward propagation of BP algorithm is used. The definition of the fitness function for particles is obtained based on the mean square deviation of BP algorithm.

IV. Simulation And Results Analysis

TABLE I. Sample And Expecting Output Of Neural Networks

Sample		Expecting Output
1	Ta-YiZhen-Feng-Shi-Di-Pao-Le	1011111000
2	ZheZhi-Ge-Tai-PingDan-WuWei-Le	0111010100
3	Ta-Cong-MaShang-XiaLai	1101100000
4	WuLi-Xue-QiLai-HenNan	0110100000
5	Ta-XueHui-Le-Jie-FangCheng	1011100000

While FPSO-based BP algorithm is used in our research, the initialized parameter values are as follows: the particle swarm size is 20, the cluster number K is 3, and the studying factors are set as c_0= 0.5, c_1=0.5, c_2=0.5, c_3=0.5. The studying ratio = 0.6, momentum factor =0.2; the evolving number is set as 3000 in order to evaluate the convergence performance of our algorithm; the single hidden layer network structure is adopted by considering the effects that the number of hidden layers causes. At the same time, the effects of node number in the hidden layer to the accuracy and convergence speed are also taken into consideration. The less the node number of the hidden layer is, the lower the network accuracy will be and the worse the fitness will be. On the contrast, the larger the node number of the hidden layer is, the longer the training time will be. Thus, the requirements of the on-line performance evaluation cannot be satisfied, even the pre-evaluation accuracy of networks would be decreased. Finally, the node number of the hidden layer is set as 60 after considering both the numbers of the input and output nodes and the number of samples, and recursively conducting comparison experiments. The training samples' expecting output and the segmentation sentence are shown in TABLE I.

In order to clarify conveniently, we name the traditional BP algorithm as Model 1, and the FPSO-based BP algorithm proposed in this paper as Model 2. The output accuracy ratios of each sample in the 5 sentence samples after being trained 3000 times, are shown in TABLE II (due to the limitation of the paper, only the values of 9 output nodes are listed in TABLE II).

The analysis and comparison of the two algorithms are discussed next.

1) The comparison of the network convergence

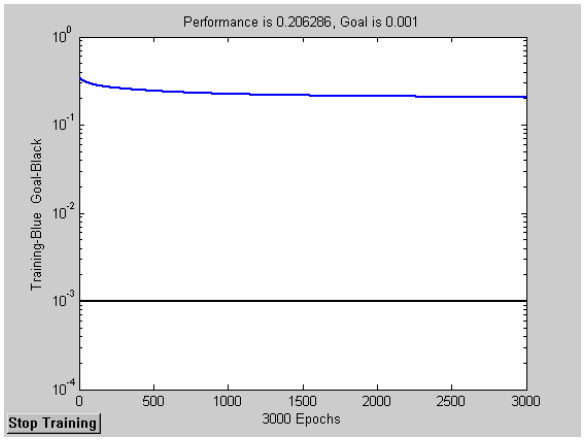

Figure 1. The error convergence curves of Model 1.

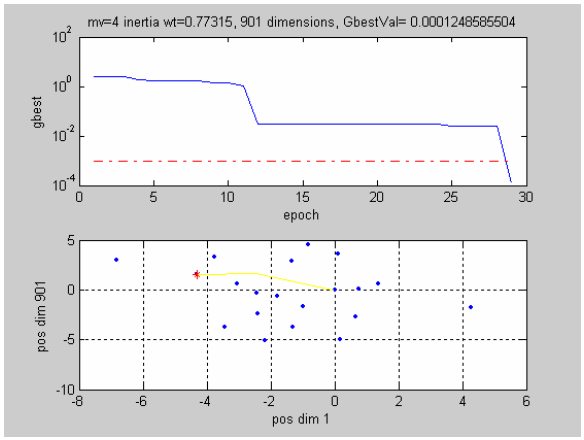

Figure 2. The error convergence curve and particle optimization of Model 2.

The training goal curve of the pure BP algorithm is shown in Fig. 1 with err_goal = 0.001 and lr = 0.01. We have that its response results TT = [0.9230 0.0606 0.9718 0.9272 0.9638 0.9171 0.9487 0.0479 0.0479] and the running time elapsed_time = 87.1230s. When the error reaches 0.5 after Model 1 has been trained 3000 times, the error curve has the intendance of no change and the network error almost has no effect. Thus, the convergence error of 0.5 for the algorithm can far less meet the error requirement of 0.001 set before. The error curves of the training process for Model 2 are shown in

TABLE II. OUTPUT ACCURACY RATIOS OF SAMPLES AFTER BEING TRAINED 3000 TIMES

Sample		Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	Y9
1	Model 1	0.9230	0.0606	0.9718	0.9272	0.9638	0.9171	0.9487	0.0479	0.0479
	Model 2	0.9991	0.0041	0.9998	1.0004	0.9994	1.0003	0.9996	0.0037	0.0030
2	Model 1	0.0601	0.8217	0.8941	0.9245	0.0125	0.9004	0.0354	0.8947	0.0213
	Model 2	0.0014	0.9817	0.9978	0.9916	0.0023	0.9991	0.0045	0.9946	0.0008
3	Model 1	0.9501	0.8875	0.0092	0.8714	0.9230	0.0631	0.0651	0.0651	0.0651
	Model 2	0.9978	0.9821	0.0028	0.9991	0.9917	0.0008	0.0009	0.0009	0.0009
4	Model 1	0.0786	0.8185	0.9031	0.1239	0.8831	0.1002	0.1342	0.0451	0.0981
	Model 2	0.0009	0.9902	0.9997	0.0072	0.9974	0.0056	0.0091	0.0031	0.0078
5	Model 1	0.9501	0.0375	0.8092	0.8714	0.9230	0.0431	0.0451	0.0651	0.0651
	Model 2	0.9932	0.0078	0.9487	1.0009	0.9814	0.0054	0.0006	0.0003	0.0003

TABLE III. OUTPUT RELATIVE ERROR OF DIFFERENT MODELS OF SAMPLE 1

Model	Average Error	Output Relative Error (‰)								
		Y1	*Y2*	*Y3*	*Y4*	*Y5*	*Y6*	*Y7*	*Y8*	*Y9*
Model 1	58.98	80.42	67.62	29.42	77.12	27.24	90.09	51.42	42.90	42.90
Model 2	0.79	0.90	1.81	0.27	0.41	0.59	0.30	0.40	2.04	1.97

sub-graph 1 of Fig. 2. FPSO-based BP algorithm converged to the assigned accuracy ε_{BP} (ε_{BP}=0.001) after running 28 times. The total running time is elapsed_time = 39.1250s, and we have the recall result TT = [0.9991 0.0041 0.9998 1.0004 0.9994 1.0003 0.9996 0.0037 0.0030]. Obviously, the iteration number to train networks when FPSO-based BP algorithm being used is much less than that of being used of the conventional BP algorithm. While in sub-graph 2 of Fig. 2, 20 blue nodes represent 20 particles, the red nodes with * represent global best particles, and the yellow traces show the process when particles search the optimization. From all these, we can see that the FPSO algorithm costs less time while searching the optimization solution, and is not easily be premature and stagnant.

2) Comparison of Generalization Error

Different output results for the testing data are obtained when 30 samples are tested using two algorithms. Part of the data is presented in TABLE III.

It is shown from data in TABLE III that the generalization error reaches 30‰ - 80‰ and the average error reaches 58.98‰ when testing by the traditional BP algorithm. While the generalization error reaches 0.2‰～2‰ and the average error reaches 0.79‰ when testing by the FPSO-based algorithm proposed in this paper. Therefore, the generalization performance of FPSO-based algorithm is better than that of both the inheritance-algorithm-based BP algorithm and the basic BP algorithm.

V. CONCLUSION

The principle and construction on how to improve the neural networks word segmentation model based on the neural network algorithm of improving particle swarm algorithms is discussed in the paper. Large amount of simulation conducted through MATLAB. From the experiment results, we see that the improved neural networks avoid not only the problems that BP algorithm runs into the local minimum and converges slowly, but also the problems of long searching time, slow speed, easily appearing prematurity, stagnancy, etc., that are caused when the particle swarm algorithms search for the optimization solution by using positive feedback principle. Therefore, it is very meaningful to the automatic processing of Chinese information that the practicability and efficiency of word segmentation of our model can be increased further.

REFERENCES

[1] Yinfeng, Design and analysis of Chinese word segmentation automation system based on neural network, Journal of the China Society for Scientific and Technical Information (in Chinese), 1998,1,pp. 41–49

[2] Pan Hao, Hou Qinglan, "A BP Neural Networks Learning Algorithm Research Based on Particle Swarm Optimizer", Computer, Engineering and Applications[J]. vol. 42.16, pp.41–43,2006.

[3] Lin Chen, Jia He, A Particle Swarm Optimization Based on Fuzzy C-Means Clustering, Journal of Southwest university for nationalities [J]., Vol 33, No 4, pp.739–742,2007.4

[4] Jia He, Lin Chen, Hong Xu, Chinese Word Segmentation Using Back-propagation Trained by Genetic Algorithm DCDIS A Supplement, Advances in Neural Networks, Vol.14(S1), pp.416–420,2007

[5] Bao-Yi W and Shao-Min Z, A Chinese text classification model based on vector space and semantic meaning, {\it Machine Learning and Cybernetics}, Shanghai , 2004,vol. 2,pp. 1141–1145.

[6] J.Kennedy, R.C.Eberhart. Particle swarm optimization [A]. Proceedings of the IEEE International Conference on Neural Networks[C]. 1995,pp.1942–1948.

[7] P.J.Angeline.Evolutionary Optimization Versus Particle Swarm Optimization:Philosophy and Performance Difference[R].Annual Conference Center on Evolutionary Programming,San,1998.

[8] Xianghao Li, Hongxing Li etc, fuzzy cluster analysis and its application, Guizhou science and technology press,1994,pp.27，102–110，184–286.

[9] Ying Gao, Shengli Xie etc. Multi-subpopulation optimization algorithm based on clustering [J]. application research of computers, 2006,4,pp. 40–41.

[10] R.C.Eberhart,Y.H.Shi.Evolving Artificial Neural Networks[R]. Proceedings of International Conference on Neural Networks and Brain, Beijing, 1998.

[11] Cichocki A and Unbehauen R, Neural networks for opimization and signal processing. England: John Wilcy and Sons Ltd, pp. 142–150, 1993.

[12] W. Schiffmann, M. Joost and R. Werner, Optimization of the backpropagation algorithm for training multilayer perceptrons technical Report, University of Koblenz, Institute of Physics, 1993.

[13] Hecht - Nielsen R, Theory of backpropagation neural networks , Proc. IJCNN - 89, pp. 1–593, 1989.

This work was supported by Natural Science and Technology Development Fund of CUIT (Chengdu University of Information Technology) CSRF200705.