Hindawi Publishing Corporation
The Scientific World Journal
Volume 2014, Article ID 906546, 7 pages
http://dx.doi.org/10.1155/2014/906546

Research Article

An Improved Kernel Based Extreme Learning Machine for Robot Execution Failures

Bin Li,[1,2] Xuewen Rong,[1] and Yibin Li[1]

[1] *School of Control Science and Engineering, Shandong University, Jinan, Shandong 250061, China*
[2] *School of Science, Qilu University of Technology, Jinan, Shandong 250353, China*

Correspondence should be addressed to Xuewen Rong; rongxw@sdu.edu.cn

Received 8 February 2014; Accepted 18 March 2014; Published 8 April 2014

Academic Editors: S. Balochian and V. Bhatnagar

Robot execution failures prediction (classification) in the robot tasks is a difficult learning problem due to partially corrupted or incomplete measurements of data and unsuitable prediction techniques for this prediction problem with little learning samples. Therefore, how to predict the robot execution failures problem with little (incomplete) or erroneous data deserves more attention in the robot field. For improving the prediction accuracy of robot execution failures, this paper proposes a novel KELM learning algorithm using the particle swarm optimization approach to optimize the parameters of kernel functions of neural networks, which is called the AKELM learning algorithm. The simulation results with the robot execution failures datasets show that, by optimizing the kernel parameters, the proposed algorithm has good generalization performance and outperforms KELM and the other approaches in terms of classification accuracy. Other benchmark problems simulation results also show the efficiency and effectiveness of the proposed algorithm.

1. Introduction

The reliability of robot is very important for improving the interactive ability between robot and the changing conditions. In the complex environments in which execution failures can have disastrous consequences for robots and the objects in the surroundings, the prediction ability of robot execution failures is equally important in the robotic field.

However, the prediction of robot execution failures is a difficult learning problem. The first reason is the partially corrupted or incomplete measurements of data. And the second reason is that some prediction techniques are not suitable for predicting the robot execution failures with little samples.

For improving the prediction accuracy of the robot execution failures, in 2009, Twala formulated the robot execution failures problem as a classification task that works with the probabilistic approach-decision tree for handling incomplete data [1]. In 2011, the performance of base-level and meta-level classifiers is compared by Koohi et al. and the Bagged Naïve Bayes is found to perform consistently well across different settings [2]. However, the learning techniques were not incorporated in the aforementioned studies in order to improve the prediction accuracy of robot execution failures. In 2013, Diryag et al. presented a novel method for prediction of robot execution failures based on BP neural networks [3]. The results show that the method can successfully be applied for the robot execution failures with prediction accuracy of 95.4545%. However, it is clear that the learning speed of BP neural networks is generally very slow and may easily converge to local minima. Therefore, some algorithms of machine learning field with better learning performance should be used for the robot execution failures.

The applications of neural networks are very diverse, and, in robotics, many artificial intelligence approaches are applied. Among the approaches of neural networks, extreme learning machine (ELM) proposed by Huang et al. in 2006 has fast learning speed and good generalization performance and has been used in many fields except for the robot execution failures.

The ELM is a learning algorithm for single hidden layer feed-forward neural networks (SLFNs), which determines the initial parameters of input weights and hidden biases randomly with simple kernel function. However, the stability

and the generalization performance are influenced by the above learning technique [4]. And some improvements to the ELM learning algorithm have been presented [5].

Among the influence factors of the learning performance of the ELM algorithm, the hidden neurons of the ELM learning algorithm are very important for improving generalization performance and stability of the SLFNs. In [6], we proposed an extreme learning machine with tunable activation function learning algorithm for solving the data dependent on hidden neurons. However, how to choose the suitable combination of activation functions of hidden neurons is still unresolved. In addition, when the feature mapping function of hidden neurons is unknown, kernel function can be used for improving the stability of algorithm [7], which is called the kernel based extreme learning machine (KELM). However, the kernel parameter should be chosen properly for improving the generalization performance of the KELM learning algorithm.

In order to improve the classification accuracy (generalization performance) of robot execution failures, we propose a novel kernel based extreme learning machine in this paper. The kernel parameters of kernel functions of the proposed algorithm are optimized based on the particle swarm optimization approach, which can improve the generalization performance with stable learning process of the proposed algorithm. The simulation results in terms of robot execution failures and some other benchmark problems show the efficiency and effectiveness of the proposed algorithm and are suitable for the robot execution failures problem of little (incomplete) or erroneous data.

The remainder of this paper is organized as follows. The kernel based extreme learning machine (KELM) is introduced in Section 2. Section 3 describes the particle swarm optimization for KELM learning algorithm. Then, the performance analysis of the proposed algorithm and simulation results of robot execution failures are analyzed in Section 4. Section 5 gives the performance analysis of the algorithms by two regression and two classification problems. The last section is the conclusions of this paper.

2. Kernel Based Extreme Learning Machine

Recently, the ELM learning algorithm with fast learning speed and good generalization performance has been attracting much attention from an increasing number of researchers [4, 7]. In ELM, the initial parameters of hidden layer need not be tuned and almost all nonlinear piecewise continuous functions can be used as the hidden neurons. Therefore, for N arbitrary distinct samples $\{(x_i, t_i) \mid x_i \in R^n, t_i \in R^m, i = 1, \ldots, N\}$, the output function in ELM with L hidden neurons is

$$f_L(x) = \sum_{i=1}^{L} \beta_i h_i(x) = h(x)\beta, \tag{1}$$

where $\beta = [\beta_1, \beta_2, \ldots, \beta_L]$ is the vector of the output weights between the hidden layer of L neurons and the output neuron and $h(x) = [h_1(x), h_2(x), \ldots, h_L(x)]$ is the output vector of the hidden layer with respect to the input x, which maps the data from input space to the ELM feature space [7].

For decreasing the training error and improving the generalization performance of neural networks, the training error and the output weights should be minimized at the same time, that is,

$$\text{Minimize: } \|H\beta - T\|, \|\beta\|. \tag{2}$$

The least squares solution of (2) based on KKT conditions can be written as

$$\beta = H^T\left(\frac{1}{C} + HH^T\right)^{-1} T, \tag{3}$$

where H is the hidden layer output matrix, C is the regulation coefficient, and T is the expected output matrix of samples.

Then, the output function of the ELM learning algorithm is

$$f(x) = h(x) H^T\left(\frac{1}{C} + HH^T\right)^{-1} T. \tag{4}$$

If the feature mapping $h(x)$ is unknown and the kernel matrix of ELM based on Mercer's conditions can be defined as follows:

$$M = HH^T : m_{ij} = h(x_i) h(x_j) = k(x_i, x_j), \tag{5}$$

thus, the output function $f(x)$ of the kernel based extreme learning machine (KELM) can be written compactly as

$$f(x) = [k(x, x_1), \ldots, k(x, x_N)]\left(\frac{1}{C} + M\right)^{-1} T, \tag{6}$$

where $M = HH^T$ and $k(x, y)$ is the kernel function of hidden neurons of single hidden layer feed-forward neural networks.

There are many kernel functions satisfying the Mercer condition available from the existing literature, such as linear kernel, polynomial kernel, Gaussian kernel, and exponential kernel. In this paper, we use three typical kernel functions for simulation and performance analysis and the chosen kernel functions are as follows.

(1) Gaussian kernel:

$$k(x, y) = \exp(-a\|x - y\|); \tag{7}$$

(2) hyperbolic tangent (sigmoid) kernel:

$$k(x, y) = \tanh\left(bx^T y + c\right); \tag{8}$$

(3) wavelet kernel:

$$k(x, y) = \cos\left(d\frac{\|x - y\|}{e}\right)\exp\left(-\frac{\|x - y\|^2}{f}\right), \tag{9}$$

where Gaussian kernel function is a typical local kernel function and tangent kernel function is a typical global nuclear function, respectively [8]. Furthermore, the complex wavelet kernel function is also used for testing the performance of algorithms.

In the above three kernel functions, the adjustable parameters a, b, c, e, and f play a major role in the performance of neural networks and should be tuned carefully based on the solved problem.

Compared with the ELM learning algorithm, the hidden layer feature mapping need not be known and the number of hidden neurons need not be chosen in the KELM. Moreover, the KELM learning algorithm achieves similar or better generalization performance and is more stable compared to traditional ELM and it is faster than support vector machine (SVM) [7, 9].

3. Particle Swarm Optimization for KELM

In KELM learning algorithm, the regulation coefficient C and kernel parameters should be chosen appropriately for improving the generalization performance of neural networks. In [7], the parameters are tried in a wide range and are time consuming. And in [10], a hybrid kernel function is proposed for improving the generalization performance of KELM. However, how to choose the optimal value of the parameters of kernel function has not been resolved.

In this paper, an optimization approach is introduced to the KELM for choosing the optimal parameters of kernel function. There are many optimization approaches in machine learning field and, compared with other methods, the particle swarm optimization (PSO) is a biologically inspired computational stochastic optimization technique developed by Eberhart and Kennedy [11]. The PSO approach is becoming popular because of its little memory requiring, simplicity of implementation, and ability to converge to a reasonably optimal solution quickly [12].

Similar to other population based optimization approaches, the PSO algorithm works by initialing the population of individuals randomly in the search space. Each particle of PSO can fly around to find the best solution in the search space; meanwhile, the particles all look at the best solution (best particle) in their path.

Suppose that the dimension of search space of PSO is D and the population size is $\widehat{N}$. Then, x_i^d and v_i^k are denoted by the current position and the current velocity of ith particle at iteration t, respectively. Then, the new velocity and position of the particles for the next iteration time are calculated as follows:

$$v_i^k(t+1) = w \cdot v_i^k(t) + c_1 \cdot \text{rand}() \left(p_i^k(t) - x_i^k(t)\right) + c_2 \cdot \text{rand}() \left(g_i^k(t) - x_i^k(t)\right), \tag{10}$$

$$x_i^k(t+1) = x_i^k(t) + v_i^k(t+1), \quad 1 \le i \le \widehat{N}, \quad 1 \le k \le D, \tag{11}$$

where p_i^k denotes the best position of the ith particle during the search process until now, g_i^k represents the global best position, which constitutes the best position found by the entire swarm until now, w is the inertia weight, c_1 and c_2 are the acceleration constants, and rand() is a random number between 0 and 1.

In PSO algorithm, the inertia weight w maintains the expansion ability of exploring new areas in the search space. Therefore, in order to ensure higher exploring ability in the early iteration and fast convergence speed in the last part iteration, the parameter w can reduce gradually at the generation increases and is calculated as [13]

$$w(t) = w_{\max} - \text{iter} \times \frac{(w_{\max} - w_{\min})}{\max \text{ iter}}, \tag{12}$$

where $w_{\max}$ and $w_{\min}$ are the initial inertial weight and the final inertial weight, respectively. The parameter max iter is the maximum searching iteration number and iter is the iteration number at the present, respectively.

In addition, in order to enhance the global search in the early part iteration, to encourage the particles to converge to the global optimal solution, and to improve the convergence speed in the final iteration period [12, 14], the acceleration parameters c_1 and c_2 are described as

$$c_1 = (c_{1\,\min} - c_{1\,\max}) \frac{\text{iter}}{\max \text{ iter}} + c_{1\,\max}, \tag{13}$$

$$c_2 = (c_{2\,\max} - c_{2\,\min}) \frac{\text{iter}}{\max \text{ iter}} + c_{2\,\min}, \tag{14}$$

where $c_{1\,\max}$ and $c_{1\,\min}$ are the initial acceleration constant and the final acceleration constant of c_1 and $c_{2\,\min}$ and $c_{2\,\max}$ are the initial acceleration constant and the final acceleration constant of c_2, respectively. Therefore, by changing the acceleration coefficients with time, the cognitive component is reduced and the social component is increased in (10), respectively.

Based on the optimization technology of PSO with self-adaptive parameters w and c, the parameters of kernel functions of KELM are optimized for improving the generalization performance. Since the number of parameters of kernel functions is different, the dimension of the particle of the proposed algorithm in this paper also changes with the different kernel functions. Therefore, the particle (individual) θ of search space in the proposed algorithm can be defined as

$$\begin{aligned} &\theta \in [a] \text{ for Gaussian kernel,} \\ &\theta \in [b, c] \text{ for tangent kernel,} \\ &\theta \in [d, e, f] \text{ for wavelet kernel, respectively.} \end{aligned} \tag{15}$$

Thus, the kernel parameter optimization strategy of KELM based on the PSO (which is called the AKELM (adaptive kernel based extreme learning machine) learning algorithm) is summarized as follows.

Given the type of the kernel function, the training set $\{(x_i, t_i) \mid x_i \in R^n, t_i \in R^m, i = 1, \ldots, N\}$, and the initial value of regulation coefficient C, consider the following steps.

Step 1. Initiate the population (particle) based on the kernel function and the velocity and position of each particle.

Step 2. Evaluate the fitness function of each particle (the root means standard error for regression problems and the classification accuracy for classification problems).

Table 1: Feature information and class distribution of the robot execution failures.

Datasets	Instances	Classes
LP1	88	4 (1 = 24%; 2 = 19%; 3 = 18%; 4 = 39%)
LP2	47	5 (1 = 43%; 2 = 13%; 3 = 15%; 4 = 11%; 5 = 19%)
LP3	47	4 (1 = 43%; 2 = 19%; 3 = 32%; 4 = 6%)
LP4	117	3 (1 = 21%; 2 = 62%; 3 = 18%)
LP5	164	5 (1 = 27%; 2 = 16%; 3 = 13%; 4 = 29%; 5 = 16%)

Step 3. According to formulas (10)–(14), the velocity and position of the particle are modified.

Step 4. Step 2 and Step 3 are iterated repetitively until the maximal iteration time is satisfied.

Step 5. The optimal parameters of kernel function can be determined. Then, based on the optimized parameters, the hidden layer kernel matrix is computed.

Step 6. Determine the final output weights β in terms of the following equation: $\beta = H^T((1/C) + HH^T)^{-1}T$.

4. Robot Execution Failures Based on AKELM

In this paper, the AKELM learning algorithm and the KELM algorithm for robot execution failures prediction and the other benchmark problems in machine learning field are conducted in the MATLAB 7.0 with 3.2 GHz CPU and 2G RAM. The number of populations of the PSO for optimizing the kernel parameters is 200 and the maximum iteration number is 100. The initial inertial weights $w_{\max}$ and $w_{\min}$ are 0.9 and 0.4, respectively. And the initial acceleration constant values $c_{\max}$ and $c_{\min}$ are 2.5 and 0.5, respectively, which means that c_1 changes from 2.5 to 0.5 and c_2 changes from 0.5 to 2.5 over the full range of the search. The regulation coefficient C is 100 and the kernel parameters of the KELM learning algorithm are set to 1.

4.1. Data Processing. The original robot execution failures data has 90 features, which includes the evolution of forces Fx (15 samples; the following is the same), Fy, and Fz and the evolution of torques Tx, Ty, and Tz measurements on a robot after failure of detection [15].

The robot execution failures problem includes 5 datasets, each of them defining a different learning problem:

(i) LP1: failures in approach to grasp position,

(ii) LP2: failures in transfer of a part,

(iii) LP3: position of part after a transfer failure,

(iv) LP4: failures in approach to ungrasp position,

(v) LP5: failures in motion with part.

The feature information and class distribution of the robot execution failures datasets is denoted in Table 1.

As shown from Table 1, the dataset of robot execution failure has small size with 90 features and many classes with 4 for LP1, 5 for LP2, 4 for LP3, 3 for LP4, and 5 for LP5, respectively, which increases the classification difficulty of algorithms.

In [16], a set of five feature transformation strategies was defined for improving the classification accuracy. In the learning of the AKELM and KELM algorithms in neural networks, in order to ensure that different units of data have the same influence on the algorithm, the original data need to be normalized. In this paper, the data is normalized to the interval $[-1, +1]$ and can be described by the following equation:

$$x = 2\frac{x - x_{\min}}{x_{\max} - x_{\min}} - 1, \tag{16}$$

where $x_{\max}$ and $x_{\min}$ represent the maximum and minimum values in the original datasets, x on the left of the above equation is the original data, and x on the right of the above equation is the normalized output data.

For improving the generalization of the robot execution failures data, the positions of samples in each dataset are changed randomly. Then, 90% of samples of the dataset are used for training the neural networks, and the other 10% are testing samples.

4.2. Simulation and Performance Analysis. In this study, the performance of the proposed AKELM learning algorithm is compared with the KELM using the robot execution failures data. In the KELM learning algorithm, the learning ability and the generalization performance are influenced mainly by the kernel parameters of different kernel functions. In this paper, the Gaussian kernel function, tangent kernel function, and wavelet kernel function are used to construct different classifier for predicting the robot execution failures.

Firstly, in order to reduce the search space and accelerate the convergence speed of the PSO algorithm, this paper gives the relationship between the classification accuracy and the number of some parameters of kernel function on robot execution failures using the LP1 dataset. As shown in Figure 1, the classification accuracy in the interval $(0, 4]$ has good performance with the difference of the parameters 1 (the values are a, b, and d for Gaussian kernel, tangent kernel, and wavelet kernel, resp.), the parameters 2 (the values are c and e for tangent kernel and wavelet kernel, resp.), and the parameters 3 (the value is f for wavelet kernel). Therefore, the search space of the PSO algorithm is set in the interval between 0 and 4.

Since the simulation results are the same for different running times of the AKELM algorithm and the KELM algorithm, Table 2 shows the comparison of classification

Table 2: Classification accuracy of robot execution failures based on KELM and AKELM algorithms.

Kernel data	Gaussian		Tangent		Wavelet	
	KELM	AKELM	KELM	AKELM	KELM	AKELM
LP1	**100%**	**100%**	62.50%	**87.50%**	**100%**	**100%**
LP2	57.14%	57.14%	57.14%	**85.71%**	57.14%	**85.71%**
LP3	57.14%	**71.43%**	57.14%	**85.71%**	57.14%	**100%**
LP4	75%	**100%**	75%	**83.33%**	83.33%	**100%**
LP5	57.14%	**64.29%**	0%	**78.57%**	50%	**71.43%**

Table 3: Specification of benchmarks of regression and classification problems.

Datasets	Names	Attributes	Classes	Training data	Testing data
Regression	Box and Jenkins gas furnace data	10	1	200	90
	Auto-Mpg	7	1	320	78
Classification	Wine	13	3	150	28
	Diabetes	8	2	576	192

Table 4: Comparison of performance by AKELM and KELM learning algorithms for the regression problems.

Algorithms with different kernel functions	Box and Jenkins gas furnace data			Auto-Mpg		
	Training error	Testing error	Training time (seconds)	Training error	Testing error	Training time (seconds)
KELM (parameters = 1, Gaussian)	0.0120	0.0188	0.0394	0.0529	0.0599	0.1213
KELM (parameters = 1, tangent)	0.0627	0.0655	0.0116	0.6680	0.7756	0.0346
KELM (parameters = 1, wavelet)	0.0121	0.0206	0.0177	0.0509	**0.0597**	0.0415
KELM (parameters = 10, Gaussian)	0.0183	0.0213	0.0149	0.0685	0.0732	0.0286
KELM (parameters = 10, tangent)	0.2245	0.1986	0.0044	0.2071	0.2085	0.0261
KELM (parameters = 10, wavelet)	0.0306	0.0382	0.0101	0.0662	0.0712	0.0360
AKELM (Gaussian)	0.0133	**0.0183**	26.1250	0.0503	**0.0597**	74.7656
AKELM (tangent)	0.0223	**0.0242**	25.2500	0.0735	**0.0735**	73.8906
AKELM (wavelet)	0.0133	**0.0183**	28.3906	0.0502	**0.0597**	84.9688

results of robot execution failures datasets with three different kernel functions in one running time. As can be seen from the table, the proposed AKELM learning algorithm shows better classification accuracy than the KELM with different kernel functions in most cases and the best classification accuracies are given in boldface. Especially in the LP1 dataset, the proposed algorithm has 100% classification accuracy with Gaussian and wavelet kernel functions, and the generalization performance is better than the best classification approach, Bagged Naïve Bayes in [2], until now to the authors' best knowledge.

5. Performance Analysis of AKELM Using Other Benchmark Problems

In this section, the performance of AKELM learning algorithm is compared with the KELM in terms of two regression benchmarks and two classification benchmarks. Specification of the benchmark problems is shown in Table 3. The performance of classification benchmark problems is measured by the classification accuracy and the root mean squares error is used to measure the error of the regression benchmark problems.

Tables 4 and 5 show the performance comparison of AKELM and KELM with Gaussian kernel, tangent kernel, and wavelet kernel neurons; apparently, better test results are given in boldface. The parameters = 1 and parameters = 10 represent the total kernel parameters of different kernel functions set to 1 and 10, respectively.

It can be seen that the proposed AKELM algorithm can always achieve similar or better generalization performance than KELM with different kernel functions and kernel parameters. Moreover, seen from Tables 4 and 5, the KELM learning algorithm with different kernel functions has obviously different generalization performance. However, the proposed AKELM learning algorithm has similar generalization performance to different kernel functions, which means that the proposed algorithm has stable performance with kernel parameters optimized by means of the PSO algorithm, although searching the optimal parameters needs some time as the training time shown in Tables 4 and 5.

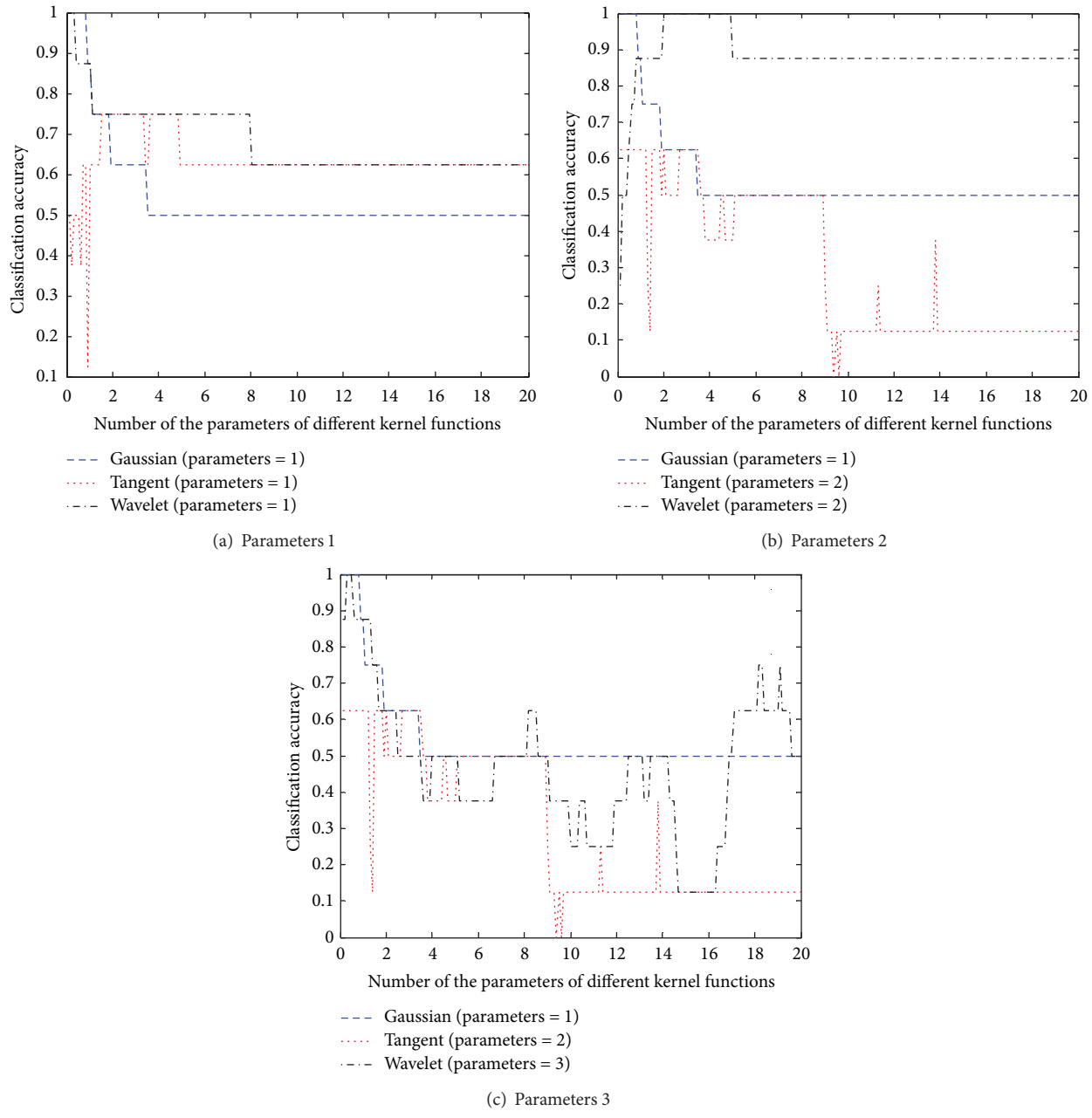

Figure 1: Relationship between the classification accuracy and the number of some parameters of kernel function on LP1 dataset.

6. Conclusions

In this study, a novel learning algorithm AKELM has been developed based on the KELM learning algorithm and the PSO approach with self-adaptive parameters. In the proposed AKELM learning algorithm, the parameters of kernel functions of neural networks are adjusted for searching the optimal values by the PSO algorithm.

As shown from the simulation results, the generalization performance of the proposed algorithm in terms of the robot execution failures datasets was found to be significantly improved compared to the KELM learning algorithm. And the other benchmark of regression and classification problems also shows that the proposed algorithm can achieve better generalization performance and has more stable ability than KELM algorithm.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China (61233014 and 61305130), the Shandong Provincial Natural Science Foundation (ZR2013FQ003

Table 5: Comparison of performance by AKELM and KELM learning algorithms for the classification problems.

Algorithms with different kernel functions	Wine			Diabetes		
	Training accuracy	Testing accuracy	Training time (seconds)	Training accuracy	Testing accuracy	Training time (seconds)
KELM (parameters = 1, Gaussian)	100%	**100%**	0.0277	84.38%	77.08%	0.1394
KELM (parameters = 1, tangent)	51.33%	50%	0.0067	73.78%	73.44%	0.1326
KELM (parameters = 1, wavelet)	100%	**100%**	0.0070	86.81%	76.56%	0.1347
KELM (parameters = 10, Gaussian)	100%	**100%**	0.0083	78.99%	79.17%	0.0919
KELM (parameters = 10, tangent)	39.33%	42.86%	0.0023	65.80%	65.63%	0.0904
KELM (parameters = 10, wavelet)	100%	96.43%	0.0061	80.03%	77.08%	0.1361
AKELM (Gaussian)	100%	**100%**	17.8594	90.45%	**80.21%**	260.7031
AKELM (tangent)	97.33%	**100%**	13.9375	73.26%	**79.17%**	313.8750
AKELM (wavelet)	100%	**100%**	16	89.06%	**79.69%**	335.5469

and ZR2013EEM027), and China Postdoctoral Science Foundation (2013M541912).

References

[1] B. Twala, "Robot execution failure prediction using incomplete data," in *Proceedings of the IEEE International Conference on Robotics and Biomimetics*, pp. 1518–1523, Guilin, China, 2009.

[2] T. Koohi, E. Mirzaie, and G. Tadaion, "Failure prediction using robot execution data," in *Proceedings of the 5th Symposium on Advances in Science and Technology*, pp. 1–7, Mashhad, Iran, 2011.

[3] A. Diryag, M. Mitic, and Z. Miljkovic, "Neural networks for prediction of robot failures," *Journal of Mechanical Engineering Science*, 2013.

[4] G.-B. Huang, D. H. Wang, and Y. Lan, "Extreme learning machines: a survey," *International Journal of Machine Learning and Cybernetics*, vol. 2, no. 2, pp. 107–122, 2011.

[5] R. Rajesh and J. Siva Prakash, "Extreme learning machines—a review and state-of-the-art," *International Journal of Wisdom Based Computing*, vol. 1, pp. 35–49, 2011.

[6] B. Li, Y. Li, and X. Rong, "The extreme learning machine learning algorithm with tunable activation function," *Neural Computing and Applications*, vol. 22, pp. 531–539, 2013.

[7] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 42, no. 2, pp. 513–529, 2012.

[8] I. S. Dhillon and D. S. Modha, "Concept decompositions for large sparse text data using clustering," *Machine Learning*, vol. 42, no. 1-2, pp. 143–175, 2001.

[9] W. Huang, N. Li, Z. Lin et al., "Liver tumor detection and segmentation using kernel-based extreme learning machine," in *Proceedings of the 35th Annual International Conference of the IEEE EMBS*, pp. 3662–3665, Osaka, Japan, 2013.

[10] S. F. Ding, Y. A. Zhang, X. Z. Xu, and L. N. Bao, "A novel extreme learning machine based on hybrid kernel function," *Journal of Computers*, vol. 8, no. 8, pp. 2110–2117, 2013.

[11] R. C. Eberhart and J. Kennedy, "New optimizer using particle swarm theory," in *Proceedings of the 6th International Symposium on Micro Machine and Human Science*, pp. 39–43, October 1995.

[12] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004.

[13] F. Han, H. Yao, and Q. Ling, "An improved extreme learning machine based on particle swarm optimization," *Neurocomputing*, vol. 116, pp. 87–93, 2013.

[14] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004.

[15] P. M. Murphy and D. W. Aha, "UCI Repository of Machine Learning Databases [EB/OL]," http://archive.ics.uci.edu/ml/datasets.html.

[16] L. Seabra Lopes and L. M. Camarinha-Matos, "Feature transformation strategies for a robot learning problem," *Feature Extraction, Construction and Selection*, vol. 453, pp. 375–391, 1998.