Using the Bees Algorithm with Kalman Filtering to Train an Artificial Neural Network for Pattern Classification

D T Pham and Haj A Darwish

Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering 2010 224: 885
DOI: 10.1243/09596518JSCE1004

Using the bees algorithm with Kalman filtering to train an artificial neural network for pattern classification

D T Pham[*†] and **A Haj Darwish**
Manufacturing Engineering Centre, Cardiff University, Cardiff, UK

The manuscript was received on 29 January 2010 and was accepted after revision for publication on 8 July 2010.

DOI: 10.1243/09596518JSCE1004

Abstract: The current paper presents the use of the bees algorithm with Kalman filtering to train a radial basis function (RBF) neural network. An enhanced fuzzy selection system has been developed to choose local search sites depending on the error and training accuracy of the RBF network. The paper provides comparative results obtained when applying RBF neural classifiers trained using the new bees algorithm, the original bees algorithm, and the conventional RBF procedure to an industrial pattern classification problem.

Keywords: bees algorithm, fuzzy logic, Kalman filter, neural network, pattern classification

1 INTRODUCTION

An artificial neural network (ANN) is a mathematical model of a biological nervous system. The model usually comprises a large number of units or nodes called neurons. In the most common types of ANN, these neurons are arranged into layers and connected together by adjustable weights to give the ANN the ability to learn. Deciding appropriate values for the weights of a neural network is a key task in the implementation of neural systems. This task can be solved using optimization methods such as the genetic algorithm [**1**], simulated annealing [**2**], and the bees algorithm [**3**]. The latter is an efficient optimization procedure inspired by the food-foraging behaviour of honeybees, which are known to adopt a judicious combination of global exploration and local exploitation to arrive at the best food source [**4**].

The current paper presents an enhancement to the original bees algorithm which involves combining it with Kalman filtering [**5**] and using the enhanced algorithm to adapt connection weights in a radial basis function (RBF) neural network. The purpose of adding Kalman filtering to the bees algorithm is to speed up its convergence. An application of the enhanced bees algorithm to train an RBF network for an industrial pattern classification problem, the automated visual identification of wood defects, is used to demonstrate the effectiveness of the enhanced algorithm compared with the original bees algorithm and the conventional RBF training procedure.

The paper is organized as follows. Section 2 reviews the bees algorithm and Kalman filtering. Section 3 describes the enhanced bees algorithm. Section 4 presents the results obtained by the enhanced algorithm and compares them with those obtained using the original algorithm as well as the conventional RBF procedure. Section 5 concludes the paper.

*Corresponding author: Manufacturing Engineering Centre, Cardiff University, Queen's Building, Newport Road, PO Box 925, Cardiff CF24 3AA, UK.
email: phamdt@Cardiff.ac.uk
† DT Plam is also a Visiting Professor of King Saud University (KSU) in Riyadh, Saudi Arabia.

2 THE BEES ALGORITHM AND KALMAN FILTERING

2.1 The bees algorithm

The bees algorithm is a swarm-based optimization procedure which mimics the natural foraging behaviour of honeybees to locate optima in complex search spaces. The algorithm employs a combination of global exploratory and local exploitative search techniques. In the original algorithm, the global exploration and local exploitation both implement random search. For the global search, 'scout bees' are sent randomly to different parts of the search space to look for potential solutions. For

the local exploitation, 'follower bees' are recruited to exploit those patches of the search space found by the scout bees to be more promising in terms of the quality of the solutions that they contain.

The basic form of the algorithm needs several parameters, including: the number of scout bees, n; the number of patches selected for the local search, m; the number of top ('elite') patches among the m selected patches, e; the number of follower bees to be recruited for the elite patches, nep; the number of bees to be recruited for the other $(m-e)$ selected patches, nsp; the initial size of each patch, ngh; and the stopping condition.

Figure 1 shows a flowchart of the basic bees algorithm. Figure 2 depicts a local rectangular patch in a two-dimensional search space. In Figure 2, ngh_1 and ngh_2 represent one-half of the sides of the local patch. The parameters define the size of the local search area.

In the bees algorithm, data are manipulated as a vector of floating point numbers, instead of the binary coding of the search space usually adopted in genetic algorithms. The vector gives the position of a bee and represents a sample from the search space. The position vector for a scout bee is randomly generated within the search space and that for a follower bee is randomly produced within a selected patch. Position updating in the local search part of the algorithm takes place in random jumps according to

$$x_{\text{new}} = x_{\text{old}} + \alpha \cdot ngh \tag{1}$$

where

$\alpha \in \text{uniform } (-1, +1)$ or $\alpha \in \text{normal } (-1, +1)$
x_{new} are the new coordinates of a bee
x_{old} are the most recent coordinates of a bee
ngh is the radius of the local search patch.

An enhanced version of the bees algorithm was proposed by Pham and Haj Darwish [**6**] to reduce the number of parameters needed to run the basic form of the algorithm. The reduction in the number of the parameters was achieved by using fuzzy logic in the selection of local search sites [**6**].

The bees algorithm has been used to solve a number of problems including optimization of a fuzzy logic controller [**7**], solution of the environmental/economic dispatch problem in power generation [**8**], designing of antenna arrays [**9**], search for protein structures [**10**], timetabling [**11**], and the design of mechanical elements [**12**].

2.2 Kalman filtering

The Kalman filter [**13**, **14**], which is a recursive estimator, was proposed by Kalman in 1960 to predict optimal parameters for a linear system. A more general form of the Kalman filter called the extended Kalman filter [**15**] was subsequently developed for systems with non-linear behaviours.

Consider a non-linear system represented as follows

$$\boldsymbol{x}_{n+1} = f(\boldsymbol{x}_n) + w_n \tag{2}$$

1 initialise population with random solutions

2 evaluate fitness of the population

3 while (stopping criterion not met)

//forming new population

4 select elite bees and elite sites for neighbourhood search

5 select other sites for neighbourhood search

6 recruit bees around selected sites and evaluate their fitness

7 select the fittest bee from each site

8 assign the remaining bees to search randomly and evaluate their fitness

9 end while.

Fig. 1 Pseudo-code of the basic bees algorithm

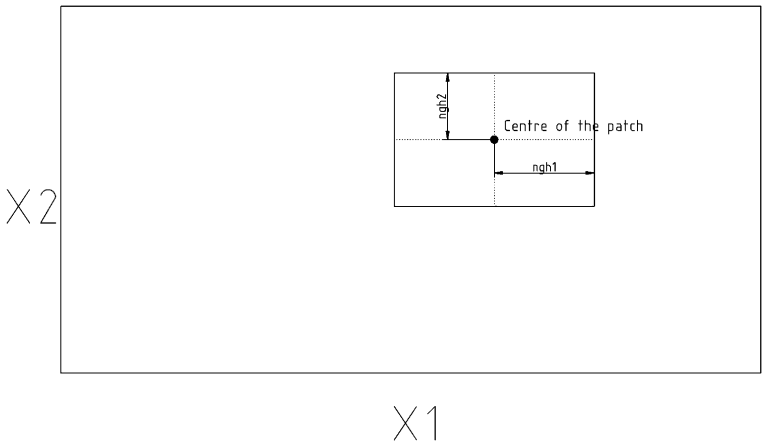

Fig. 2 A local patch in a search space

$$\boldsymbol{d}_n = h(\boldsymbol{x}_n) + \upsilon_n \tag{3}$$

where

$\boldsymbol{x}_n$ represents the state of the system at time n
w_n is the process noise
$\boldsymbol{d}_n$ is the observation vector
υ_n is the observation noise
$f(\cdot)$ and $h(\cdot)$ are non-linear vector functions of the state.

The following are the recursive estimation equations of the extended Kalman filter

$$\boldsymbol{K}_n = \mathbf{P}_n \boldsymbol{H}_n \left(\mathbf{R}_n + \boldsymbol{H}_n^{\mathrm{T}} \mathbf{P}_n \boldsymbol{H}_n\right)^{-1} \tag{4}$$

$$\hat{\boldsymbol{x}}_n = f(\hat{\boldsymbol{x}}_{n-1}) + \boldsymbol{K}_n[\boldsymbol{d}_n - h(\hat{\boldsymbol{x}}_{n-1})] \tag{5}$$

$$\mathbf{P}_{n+1} = \boldsymbol{F}_n \left(\mathbf{P}_n - \boldsymbol{K}_n \boldsymbol{H}_n^{\mathrm{T}} \mathbf{P}_n\right) \boldsymbol{F}_n{}^{\mathrm{T}} + \mathbf{Q}_n \tag{6}$$

where

$$\boldsymbol{F}_n = \left.\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_n} \tag{7}$$

$$\boldsymbol{H}_n^{\mathrm{T}} = \left.\frac{\partial h(\boldsymbol{x})}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_n} \tag{8}$$

and

$\boldsymbol{K}_n$ is the Kalman gain
$\mathbf{R}_n$ and $\mathbf{Q}_n$ are the covariance matrices of noise processes w_n and υ_n, respectively
$\mathbf{P}_n$ is the covariance of the prediction error
$\hat{\boldsymbol{x}}_n$ is the estimated state of the system at time n.

Kalman filtering has been used for tuning fuzzy systems [**15**], estimating locations in Global Positioning Systems (GPS) [**16**], and training neural networks [**17**, **18**]. However, Kalman filtering is very sensitive to the choice of starting point and to parameter tuning, as it is difficult to find proper parameters without extensive trials. Another problem in employing the Kaman filter as an optimization tool is trapping at local optima, as the filter tends to converge to local solutions quickly.

In this work, the fast convergence of the Kalman filter to local optima is exploited to construct an efficient method to update the positions of follower bees in the local search part of the bees algorithm.

3 THE PROPOSED METHOD

3.1 Enhanced fuzzy selection

In reference [**6**], a fuzzy greedy selection system was proposed to choose local search sites and to decide on the number of recruited followers for each selected patch. In this section, an enhanced fuzzy selection system is described which performs the selection and recruitment processes using multiple independent criteria. The system attempts to select patches with low training errors and high classification accuracies to yield good neural classifiers.

The selection system consists of two fuzzy inputs, two crisp outputs, and an inference system of the zero-order Sugeno type. The first input variable is the classification error and the second input variable is the training accuracy. Each input variable is represented by two triangular membership functions, namely 'low' and 'high'. Figure 3 depicts the membership functions used for the input variables. The output is made up of two quantities:

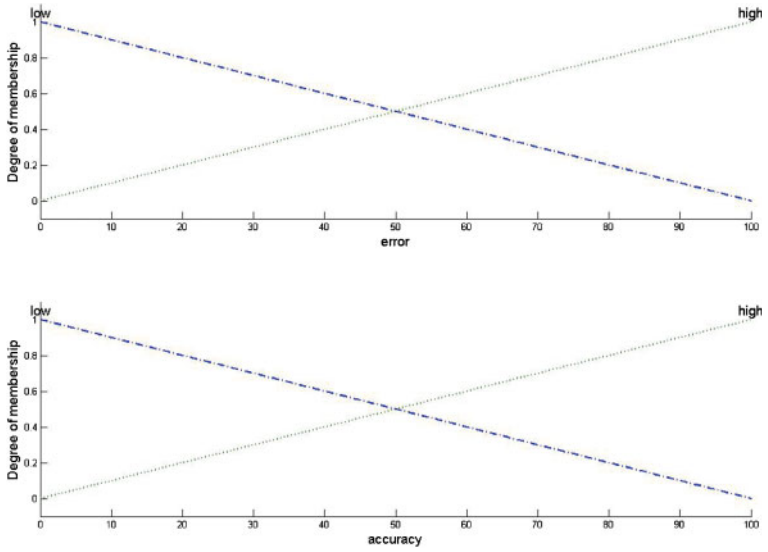

Fig. 3 Input membership functions for enhanced fuzzy selection

'low' with value zero (0) and 'high' with value (*nw*). *nw* is the maximum number of follower bees per patch.

The initial universe of discourse for the inference system is defined by the maximum and the minimum values for the classification error and training accuracy of the scout bees. The universe of discourse is updated at the end of each iteration. This repeated updating makes the selection procedure dynamic and ensures that the number of recruited bees is always appropriate for any given situation.

The selection is performed using the fuzzy rules shown in Figure 4. The structure of the rules gives the system its multi-criteria selection behaviour, since recruitment depends on two independent terms, i.e. classification error and training accuracy.

The output is rounded off to give the total number of follower bees in a selected patch. In this type of selection, there is no need to sort local search sites into a candidate list as ranking does not play any role in the fuzzy multi-criteria selection.

3.2 The algorithm

Equation (1) is a simplified form of equation (5), the recursive estimation or state update of the Kalman filter which, when linearized, can be written as

$$\hat{\boldsymbol{x}}_n = \hat{\boldsymbol{x}}_{n-1} + \boldsymbol{K}_n \boldsymbol{E}_{n-1} \tag{9}$$

where $\boldsymbol{E}_{n-1}$ is the Kalman estimation error and can be likened to the patch radius *ngh* of the bees algorithm.

Thus, the Kalman filter equation for state update, equation (5), can be used instead of equation (1) to change the positions of follower bees in the exploitation stage. It is assumed that all bees have their own memories to store the most recent values of their Kalman filter parameters.

In addition to this replacement of random jumps with Kalman filter state updating, the enhanced fuzzy selection method discussed in the section 3.1 is also employed to recruit follower bees and to choose local search sites.

1. If (classification error is high) and (training accuracy is high) then (recruitment is low)
2. If (classification error is high) and (training accuracy is low) then (recruitment is low)
3. If (classification error is low) and (training accuracy is low) then (recruitment is low)
4. If (classification error is low) and (training accuracy is high) then (recruitment is high)

Fig. 4 Enhanced fuzzy selection rules

The flowchart of the proposed method is presented in Figure 5. As with the standard bees algorithm, the modified algorithm starts in step 1 with (*ns*) scout bees being placed uniformly randomly in the search space. The fitness of each site visited by the scout bees is evaluated (i.e. the classification error and training accuracy of a neural network are calculated) in step 2.

In step 3, an enhanced fuzzy system is formed and initialized with the values of classification errors and training accuracies of the patches visited by the scouts.

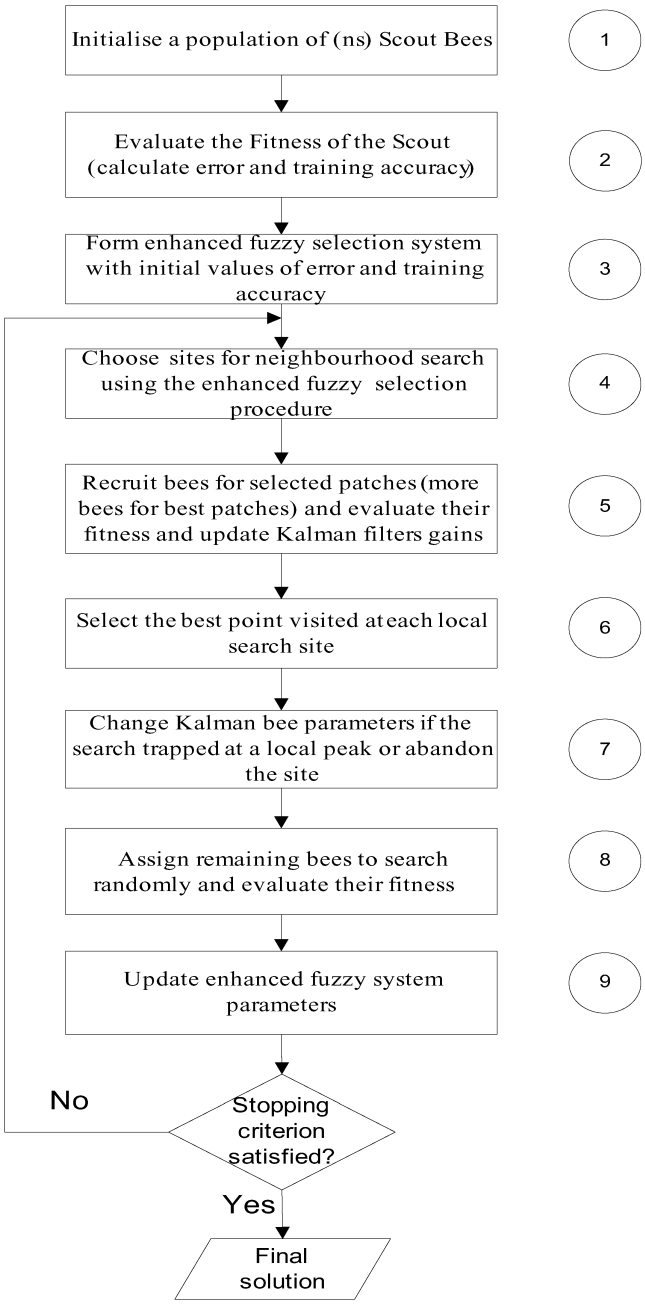

Fig. 5 Flowchart of the proposed algorithm with enhanced fuzzy selection

The best sites are selected for exploitation (local search) in step 4 and bees are recruited for those sites in step 5. Site selection and bee recruitment are performed using the enhanced fuzzy selection procedure and are conducted according to the classification error and training accuracy associated with each site (more bees for lower error and higher training accuracy). Site selection and bee recruitment are implemented smoothly by applying fuzzy rules. In step 5, the fitness values of the points visited by the recruited bees are evaluated and the Kalman filter parameters (the filter gains) for those bees are updated.

Step 6 involves ranking the points visited at each site and selecting the point with the highest fitness value to compete for further exploitation in the next iteration.

The optional step 7 is invoked when the optimization process is deemed to be trapped at a local peak, in which case the Kalman filter parameters for the associated bees are changed (in this work, this involved doubling the values of the Kalman filter parameters), or when a fitness plateau is detected, which causes stopping of exploitation at that site and abandonment of the site for a new location in the search space.

In step 8, unused scout bees (i.e. those not already 'working' at the points selected in step 6) are again sent randomly to explore the search space looking for other potential solutions.

In step 9, the new sites found by the scout bees with the points selected in step 6 are used to update the enhanced fuzzy selection system. The process is repeated from step 4 until a stopping criterion is met.

4 IDENTIFICATION OF WOOD DEFECTS

Automated visual inspection systems for identifying wood defects using neural networks are described in references [**19**] and [**20**]. Figure 6 gives examples of the twelve types of wood veneer defect and an example of clear wood. Photographs of the different types of defects and of good veneer were captured and processed to produce a dataset comprising 232 patterns, each containing 17 features [**20**]. The patterns were divided into two subsets, one used for training neural classifiers and the other for testing the trained classifiers.

An RBF neural network was used as classifier. It was configured in three layers: an input layer with 17 neurons, a hidden layer with 51 neurons, and an output layer with 13 neurons. The number of neurons in the input layer matched the number of features (or the dimension of the input pattern $\boldsymbol{x}$). The number of neurons in the output layer was the same as that of pattern classes (12 classes of defects plus the class

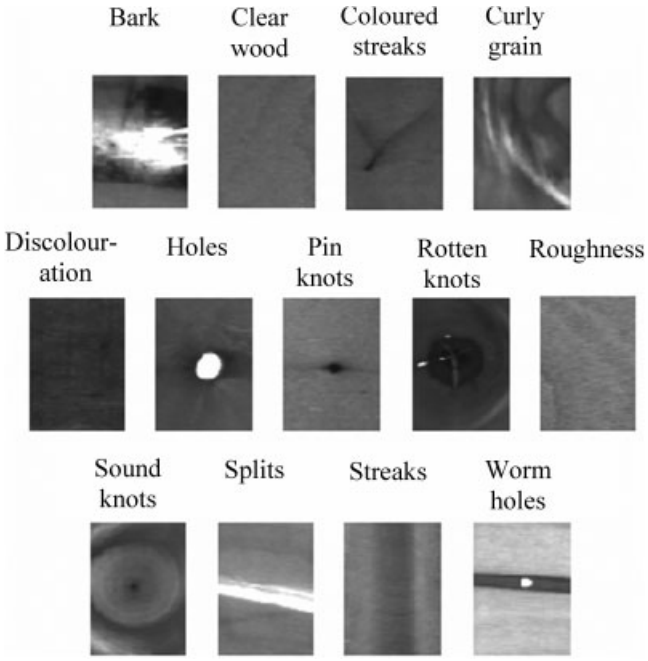

Fig. 6 Wood veneer defect types [**4**]

of clear wood). The size of the hidden layer was empirically determined.

Each of the neurons in the hidden layer applies an activation function which is a function of the Euclidean distance between the input and a 17-dimensional prototype vector. Each hidden neuron contains its own prototype vector. The outputs of the neurons in the output layer are weighted sums of the outputs of the hidden layer neurons [**21**].

The activation function (basis function) used in the hidden layer is given by [**21**]

$$g(\boldsymbol{v}) = [g_o(\boldsymbol{v})]^{1/(1-p)} \tag{10}$$

$$g_o(\mathbf{v}) = \text{norm}(\boldsymbol{x} - \boldsymbol{v})^2 \tag{11}$$

where

$p > 0$ is a real number
$\boldsymbol{x}$ is an input pattern
$\boldsymbol{v}$ is a prototype vector and the centre of a basis function.

The output of the RBF network, $\boldsymbol{y}$, is given by

$$\boldsymbol{y} = \boldsymbol{K}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}} \tag{12}$$

where $\boldsymbol{K}$ is the output of the hidden neurons and $\mathbf{W}$ is the matrix of the weights of the links between the hidden layer and the output layer.

The elements of the weight matrix $\mathbf{W}$ and those of the prototypes $\boldsymbol{v}$ define the state of a non-linear system while the output of the RBF network forms the output of that non-linear system (see equations (14) and (15))

$$\boldsymbol{x} = \begin{bmatrix} w_1^{\mathrm{T}} & \ldots & w_n^{\mathrm{T}} & v_1^{\mathrm{T}} & \ldots & v_c^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{13}$$

$$\boldsymbol{x}_{n+1} = f(\boldsymbol{x}_n) + w_n \tag{14}$$

$$\boldsymbol{y}_n = h(\boldsymbol{x}_n) + v_n \tag{15}$$

where

w_n and v_n are artificially added noise processes
$f(.)$ is the identity mapping
$\boldsymbol{y}_n$ is the target output of the RBF network
$h(\boldsymbol{x}_n)$ is the actual output of the RBF network.

The vector $\boldsymbol{x}$ consists of $(n(c+1)+mc)$ RBF parameters arranged in a linear array, where c is the number of hidden neurons (51), n is the number of output neurons (13), and m is the number of input neurons (17).

The training set consisted of 80 per cent of patterns (185 in total) selected randomly from the dataset and the remaining 20 per cent (47 in total) formed the test set. Table 1 provides details of the pattern classes and the numbers of examples used for training and testing.

Experiments were repeated ten times with ten iterations each. The number of scouts, *ns*, was five and the maximum number of follower bees in each patch, *nw*, was also five. These values were determined empirically. The covariance matrices of the Kalman filter were $\mathbf{P} = \mathbf{Q} = 10 \cdot \mathbf{I}_{1543}$, where $\mathbf{I}$ is the identity matrix and 1543 is the size of vector $\boldsymbol{x}$ (see equation (13)) and $\mathbf{R} = 10 \cdot \mathbf{I}_{2405}$, where 2405 is the

Table 1 Pattern classes and the number of examples used for training and testing

Pattern class	Total	Used for training	Used for testing
Bark	20	16	4
Clear wood	20	16	4
Coloured streaks	20	16	4
Curly grain	16	13	3
Discolouration	20	16	4
Holes	8	6	2
Pin knots	20	16	4
Rotten knots	20	16	4
Roughness	20	16	4
Sound knots	20	16	4
Splits	20	16	4
Streaks	20	16	4
Wormholes	20	16	4
Total	232	185	47

number of output neurons multiplied by the number of patterns in the training set.

The average number of evaluations needed to obtain good classification results was 115 objective function calls instead of the 100 000 iterations and the large bee population required by the standard bees algorithm (see Table 2) [**22**]. The latter was implemented in C++. Due to the large numbers of iterations required by the standard algorithm to obtain well-trained neural classifiers, the code was executed on a parallel-computing Condor pool comprising more than 1000 personal computers with a combined maximum processing power of 1012 floating point operations per second. The new algorithm was programmed and run under the MATLAB environment on a personal computer with a dual core processor and the Microsoft® Windows XP operating system. Although the use of very different hardware and software environments makes it unreasonable directly to compare the speeds of the two algorithms, an idea of the remarkable improvement achieved can be gained from the fact that the standard algorithm took approximately 2.5 days to train a classifier while the new algorithm required only 4 h. Table 3 presents a comparison of the results obtained by RBF classifiers trained conventionally and trained using the standard bees algorithm and the version with Kalman filtering. To highlight the strong performance of the bees algorithm training, the results for a minimum distance classifier (MDC) [**20**] are also provided.

Table 2 The parameters of the standard bees algorithm [**22**]

Parameter	Symbol	Value
Population	n	250
Number of selected sites	m	15
Number of elite sites out of m selected sites	e	3
Initial patch size	ngh	0.1
Number of bees for elite sites	nep	80
Number of bees for other selected sites	nsp	50

Table 3 Comparison with conventional RBF training, MDC, and the standard bees algorithm (the results for the bees algorithm with Kalman filtering were the averages for ten runs)

Pattern recognition	Training accuracy (%)	Test accuracy (%)
RBF (MATLAB) [**22**]	–	76.43
MDC [**21**]	–	63.12
RBF (standard bees algorithm) [**22**]	86.90	75.12
RBF (proposed algorithm)	91.08	78.51

5 CONCLUSIONS

This paper has presented an enhancement to the bees algorithm and demonstrated an application of the enhanced algorithm to an industrial pattern recognition problem. The enhancement consists of integrating Kalman filtering, which is a gradient-based optimization method, with the bees algorithm, which is a swarm-based optimization technique combining random exploration and 'greedy' exploitation. The Kalman filter enables rapid migration towards good solutions, while premature convergence and sensitivity to initial positions are overcome by the swarm nature of exploration in the bees algorithm. The use of enhanced fuzzy selection has eliminated the need for the bees algorithm to rank the search sites. The application discussed in this paper is the identification of wood defects by RBF neural classifiers. The new algorithm gave the RBF neural classifier better training and test results than those of RBF classifiers trained conventionally and by the standard bees algorithm. The new algorithm also drastically reduced the number of iterations needed to train the neural classifier, speeding up the training manifold in comparison with the original algorithm.

REFERENCES

1 **Goldberg, D. E.** *Genetic algorithms in search, optimization and machine learning*, 1989 (Addison-Wesley, Reading, Massachusetts, USA).
2 **Kirkpatrick, S., Gelatt, C. D. Jr,** and **Vecchi, M. P.** Optimization by simulated annealing. *Science*, 1983, **220**(4598), 671–680.
3 **Pham, D. T., Ghanbarzadeh, A., Koç, E., Otri, S., Rahim, S.,** and **Zaidi, M.** The bees algorithm – a novel tool for complex optimisation problems. In Proceedings of 2nd Virtual International Conference on *Intelligent production machines and systems (IPROMS 2006)*, (Eds D. T. Pham, E. E. Eldukhri, and A. J. Soroka), 13–14 July 2006, pp. 454–459 (Elsevier, Oxford, UK).
4 **Pham, D. T., Soroka, A. J., Ghanbarzadeh, A., Koc, E., Otri, S.,** and **Packianather, M.** Optimising neural networks for identification of wood defects using the bees algorithm. In Proceedings of IEEE International Conference on Industrial Informatics (INDIN2006), Singapore, 16–18 August 2006, pp. 1346–1351 (IEEE).
5 **Pham, D. T.** and **Haj Darwish, A.** Optimising fuzzy membership functions using the bees algorithm with Kalman filtering. In Proceedings of the 5th International Virtual Conference on Innovative

Production Machines and Systems *(IPROMS 2009)* (Edited by D. T. Pham, E. E. Eldukhri, and A. J. Soroka) Whittles, Dunbeath, Scotland, 16–17 July 2009, pp. 328–333.

6 **Pham, D. T.** and **Haj Darwish, A.** Fuzzy selection of local search sites in the bees algorithm. In Proceedings of the 4th International Virtual Conference on *Innovative Production Machines and Systems (IPROMS 2008)*, (Edited by D. T. Pham, E. E. Eldukhri, and A. J. Soroka) Whittles, Dunbeath, Scotland, 12–14 July 2008, pp. 391–397.

7 **Pham, D. T., Haj Darwish, A.,** and **Eldukhri, E. E.** Optimisation of a fuzzy logic controller using the bees algorithm. *Int. J. Comput. Aided Engng Technol.*, 2009, **1**(2), 250–264.

8 **Lee, J. Y.** and **Haj Darwish, A.** Multi-objective environmental/economic dispatch using the bees algorithm with weighted sum. In Proceedings of the EU–Korea Conference on *Science and technology (EKC2008)*, (Edited by Seung-Deeg Xoo) 2008, pp. 267–274 (Springer, Heidelberg).

9 **Guney, K.** and **Onay, M.** Bees algorithm for design of dual-beam linear antenna arrays with digital attenuators and digital phase shifters. *Int. J. RF Microwave Comput. Aided Engng*, 2008, **18**(4), 337–347.

10 **Bahamish, H. A. A., Abdullah, R.,** and **Abdul Salam, R. A.** Protein conformational search using bees algorithm. In Proceedings of 2nd IEEE Asia International Conference on *Modeling & simulation (AICMS 08)*, (Edited by D. AI-Dabass, S. Turner, G. Tan, and A. Abraham) Kuala Lumpur, 13–15 May 2008, pp. 911–916 (IEEE).

11 **Lara, C., Flores, J.,** and **Calderón, F.** Solving a school timetabling problem using a bee algorithm. In Proceedings of the 7th Mexican International Conference on *Artificial intelligence: Advances in artificial intelligence*, (Edited by A. Gelbukh and E. F. Morales), Arizapánde Zaragozo, Mexico, 27–31 October 2008, pp. 664–674 (Springer, Mexico).

12 **Pham, D. T., Ghanbarzadeh, A., Otri, S.,** and **Koç, E.** Optimal design of mechanical components using the bees algorithm. *Proc. IMechE, Part C: J. Mechanical Engineering Science*, 2009, **223**(C5), 1051–1056. DOI: 10.1243/09544062JMES838.

13 **Kalman, R. E.** A new approach to linear filtering and prediction problems. *Trans. ASME J. Basic Engng*, 1960, **82**(Series D), 35–45.

14 **Russell, S.** and **Norvig, P.** *Artificial intelligence: A modern approach*, 2004 (Prentice-Hall, Upper Saddle River, New Jersey, USA).

15 **Nian, Z.** and **Wunsch, D. C.** An extended Kalman filter (EKF) approach on fuzzy system optimization problem. In Proceedings of the 12th IEEE International Conference on Fuzzy Systems, St. Louis, Missouri, USA, 25–28 May 2003, pp. 1465–1470 (IEEE).

16 **Grewal, M. S., Weill, L. R.,** and **Andrews, A. P.** *Global positioning systems, inertial navigation, and integration*, 2007 (Wiley, Hoboken, New Jersey, USA).

17 **Simon, D.** Training radial basis neural networks with the extended Kalman filter. *Neurocomputing*, 2002, **48**(1–4), 455–475.

18 **Wang, J., Zhu, L., Cai, Z., Gong, W.,** and **Lu, X.** Training RBF networks with an extended Kalman filter optimized using fuzzy logic. In Proceedings of IFIP TC12 International Conference on *Intelligent information processing*, (Edited by Z. Shi, K. Shimohara, and D. Feng), Adelaide, 20–23 September 2006, pp. 317–326 (Springer, Boston).

19 **Pham, D. T.** and **Alcock, R. J.** Automatic detection of defects on birch wood boards. *Proc. IMechE, Part E: J. Process Mechanical Engineering*, 1996, **210**(15), 45–52. DOI: 10.1243/PIME_PROC_1996_210_292_02.

20 **Packianather, M. S.** and **Drake, P. R.** Identifying defects on plywood using a minimum distance classifier and a neural network. In Proceedings of the 1st Virtual International Conference on *Intelligent production machines and systems (IPROMS 2005)*, (Edited by D. T. Pham, E. E. Eldukhri, and A. J. Soroka) 14–15 July 2005, pp. 543–548 (Elsevier, Oxford).

21 **Chen, S., Cowan, C. F. N.,** and **Grant, P. M.** Orthogonal least squares learning algorithm for radial basis function networks. *IEEE Trans. Neural Network.*, 1991, **2**(2), 302–309.

22 **Ghanbarzadeh, A.**, The bees algorithm: A novel optimisation tool. Manufacturing Engineering Centre, Cardiff University, Cardiff, UK, 2007, p. 158.