This article can be cited as Q. H. Do, Predictions of Machine Vibrations Using Artificial Neural Networks Trained by Gravitational Search Algorithm and Back-Propagation Algorithm, International Journal of Artificial Intelligence, vol. 15, no. 1, pp. 93-111, 2017.

Predictions of Machine Vibrations Using Artificial Neural Networks Trained by Gravitational Search Algorithm and Back-Propagation Algorithm

Quang Hung Do

Faculty of Information Technology, University of Transport Technology,
Hanoi, Vietnam
Email: hungdq@utt.edu.vn; quanghung2110@gmail.com

ABSTRACT

Since the vibration is the main factor to result in machine faults, predictions of machine vibrations are necessary for improving operational efficiency, product quality, and safety. This raises the need to have an effective model that can be used to predict machine vibrations. In this study, an Artificial Neural Network (ANN) model along with Gravitational Search Algorithm (GSA) and a Back-Propagation (BP) Algorithm - Gradient Descent with Momentum (GDM) is proposed. We first identify the factors that may cause machine vibrations and then construct a dataset. The hybrid algorithm of improved GSA and GDM is then utilized to optimize the weights between layers and biases of the neural network. A real application is used to illustrate the applicability of the model. The results show that the proposed approach achieves a high accuracy. The results are also compared with those obtained from the ANN-based models trained by other algorithms. The comparative analysis indicates that the proposed model performs better than the others. It is expected that the proposed model may be used in the prediction of machine vibrations and can aid in the development of a novel approach for prediction issues faced in machine tools industry.

Keywords: Artificial neural network, Gravitational Search Algorithm, machine vibrations, prediction.

Mathematics Subject Classification: 92B20

Computing Classification System: I.2.1, I.2.8

1. INTRODUCTION

Machinery plays an important role in the infrastructure of an industrial plant. When operating the machine tools, machine tools vibrate more or less. Excessive vibration may cause operational efficiency, product quality problems. Therefore, predictions of machine vibration are significant for reducing machine down time. In the past, predictions of machine vibrations were based on practical considerations, previous experiences, historical data, and common sense. Therefore, the successful prediction of vibration requires staff with considerable degree of knowledge and experiences. Routine work in data collection may be carried out by trained personnel, but data processing and assessment of the state of a machine require an engineer who has knowledge in various areas (design of machines, dynamics, mathematics, signal processing, etc.) and who is able to use this knowledge in context. Since machinery has become more complex and technologically sophisticated, traditional prediction methods have lags behind. In addition, many methods of prediction require a great deal of expertise to apply them successfully (Jack and Nandi, 2002). Hence, there has been an increasing requirement of selecting appropriate techniques with accurate prediction and a need for advanced models that can make automatic and reliable decisions on the running health of machine tools.

Artificial intelligence (AI) techniques are recognized as attractive alternatives to the standard, well-established hard computing paradigms. AI techniques, which emphasize gains in understanding system behavior, have been proven to be able to efficiently solve complicated problems. The intelligent prediction is the application of the artificial intelligence technique in the area of predicting machine conditions. AI techniques are very powerful and promising tools. One of the widely used AI models is the artificial neural network (ANN). Accordingly, research has been carried out into the use of ANN for prediction/diagnosis purposes. ANN is developed to mimic human decision-making in vibration analysis (Kumar et al., 2012). The models that use ANN can efficiently learn the status or operation conditions of the machines.

When ANNs are structured for a particular application, they must be trained before being put to use. The training phase adjusts parameters in order to minimize a cost function defined as the mean squared error (MSE) or sum of squared error (SSE) between its actual and target outputs. Presenting a satisfactory and efficient training algorithm has always been a challenging subject. A popular approach used in the training phase is the back-propagation (BP) algorithm, including the standard BP (Hush and Horne, 1993) and the improved BP (Adeli and Hung, 1994; Hagar and Menhaj, 1994; Zhang, 2009). However, researchers have pointed out that the BP algorithm - a gradient-based algorithm - has some disadvantages (Gupta and Sexton, 1999; Mirjalili et al., 2012). Heuristic algorithms are known for their ability to produce optimal or near optimal solutions for optimization problems. There have been a lot of developments in heuristic algorithms' improvement to deal with practical problems. For example, a study of Azar et al. (2016) proposed a hybrid algorithm, combining two heuristic algorithms simulated annealing and ant colony optimization, to predict software quality attributes of new unseen systems. The proposed algorithm gives promising results and is enough to apply to any software quality attributes. A genetic algorithm for optimal tuning of a linear controller was presented by Martin et al. (2009) to apply to the networked control of a high-performance drilling process, a class of complex electromechanical process. The proposed method yielded a simpler controller that performs better in a real time application. A stochastic variable neighbourhood algorithm (P. Solos et al., 2016) was proposed to solve a problem of shift scheduling of tank trucks faced by a small oil company. The optimization algorithm proposed in current contribution manages to achieve better solutions for all but one instance among thirty of them in the same reasonable amount of computational time. Precup et al. (2014) proposed a novel Adaptive Charged System Search algorithm for the optimal tuning of Takagi–Sugeno proportional–integral fuzzy controllers. The algorithm was applied to the nonlinear control of a class of servo systems characterized by second-order models with an integral component. This algorithm reduced the number of tuning parameters and simplified the optimization problems. Other than those, a heuristic algorithm - Gravitational Search Algorithm (GSA), inspired by the behavior of natural phenomena – was also developed for solving optimization problems. Through some benchmarking studies, this algorithm has been proven to be powerful and is considered to outperform other algorithms.

The merit of the GSA algorithm and the success of ANN in the prediction area have encouraged us to use this heuristic algorithm for training ANN. In this research, we propose a model based on the ANN improved by the GSA and BP hybrid algorithm for predicting machine vibrations. In addition, some improvement of the GSA was also made to improve the performance. To the best of our knowledge, the combination of these artificial intelligence techniques is applied for the first time in any application area.

The rest of this paper is organized into seven sections. The literature review is presented in Section 2. Section 3 describes the ANN. Section 4 is dedicated to the GSA. The proposed hybrid algorithm is provided in Section 5. The research design is in Section 6. Section 7 is devoted to experimental results and discussion. Finally, Section 8 presents the conclusion.

2. LITERATURE REVIEW

The ANN has been widely used in different applications (Aghajanloo et al., 2013; Gupta and Sexton, 1999; Khalaj et al., 2014; Ranković et al., 2014; Taghavifar and Mardani, 2014). The studies showed that applying neural networks to solve optimization problems is superior to conventional techniques. An optimization problem is a pair of an objective function and a set of constraints on the variables. The objective is to find the value of variables that make the value of function optimal (minimum or maximum), while meeting all the constraints. Neural networks have been used to solve a wide variety of practical optimization problems that are difficult to solve using conventional techniques, including computer vision and speech recognition, pattern classification, diet problem, character recognition, image compression, stock market prediction, traveling saleman's problem, and other economics applications.

This paragraph provides a glimpse into the literature concerning the use of ANN in the area of prediction/diagnosis of machine faults. Machine fault diagnoses are a process of judging the operation condition of equipment as well as to provide basic information for the repair and restoration of machines. Applying AI techniques to prediction/diagnosis has yielded good results (Lei et al., 2008). Among the AI techniques, the ANN has been extensively used to predict machine vibrations, due to its strong and easily attainable nonlinear approach. Su et al. (2005) proposed a model for induction machine fault detection. In the model, the ANN is trained with vibration spectra and machine faults are detected from changes in the expectation of the vibration spectra modeling error. Their model was shown to be quite effective in detecting the early stages of many frequently encountered motor faults. Barakat et al. (2013) introduced a non-parametric supervised classifier based on ANN for diagnosis issues. In their study, a parameter selection was developed for automatic fault detection and diagnosis in industrial environments. The proposed model was applied to classify experimental machinery faults of rotary elements and to detect and diagnose any disturbances in a chemical plant. The results obtained from the experiments are accurate and have the highest confidence rate in comparison to several techniques. Nyanteh and Srivastava (2013) used ANN to detect short-circuit fault currents in the stator windings of a permanent-magnet synchronous machine. In the model, particle swarm optimization (PSO) was used to adjust the weights of the ANN. The ANN-based technique was shown to be effective and can be applied to real-time fault detection. A model based on the hierarchical neural network was presented by Prieto et al. (2013) to diagnose bearing faults. Kalkat (2014) focused on developing a model based on neural networks to predict and evaluate the noise of washing machine systems. His work showed that the neural network with the quick propagation algorithm gives superior performance. Based on ANN and support-vector machine (SVM), Ye et al. (2013) proposed a smart diagnosis method that can learn from repair history and accurately localize the root cause of a failure. In the study, the ANN generated a visual relationship between syndrome and root cause, and SVM created an optimal hyper plane to separate the root cause in syndrome space. Their experiments were performed on three industrial boards in high-volume production. The results indicated that their method has a significant improvement over the traditional manual diagnosis used in production. Chen (2013) introduced a fault diagnosis system based on the neural network for the electrical system of rollers. The system produced a sample output that corresponded to the expected output, indicating that it was reliable and met the requirement of fault diagnosis. A combination of wavelet packet transformation and a BP neural network was proposed by Zhang et al. (2007) for fault diagnosis. After obtaining the fault feature, their model was utilized to analyze the nonlinear mapping relationship of the failure symptom, thus realizing the separation and classification of the fault types. Lin (2011) presented an innovative method based on the modified ARTMAP neural network to synthesize low level information such as vibration signals, with high level information like signal patterns, to form a rigorous theoretical base for condition-based predictive maintenance. In order to test the performance of the method, an extensive bearing fault experiment was conducted. The experiment results demonstrated that the method has an ability to correctly detect and identify several machine faults. Moosavian et al. (2013) conducted surveys of a new scheme for fault diagnosis of internal combustion (IC) engines based on the power spectral density (PSD) technique and two classifiers, namely the K-nearest neighbor (KNN) and ANN. Thirty features were extracted from the PSD values of signals as a feature source for fault diagnosis. KNN and ANN were trained by a training data set and then used as diagnostic classifiers. Results demonstrate that the performance of ANN is better than KNN. Khodja and Chetate (2005) used ANN to construct a diagnostic system to find different defects by classification. The extracted features entered into the ANN provided recognition and identification, and then generated a healthy index that indicates the health condition of the machine. In Liao's study (2013), a method based on local mean decomposition and the neural network was applied to the fault diagnosis of rolling bearings. The kurtosis coefficient and energy characteristic parameters were extracted to form a fault feature vector, which was used as the input parameters of the BP neural network. Their work produced a well-trained neural network that can be used for classification recognition. Real applications showed that this method can accurately and effectively identify fault types of rolling bearings, and has a higher failure recognition rate than the method based on wavelet packet analysis and neural networks. Chen et al. (2016) proposed an ANN-based model to classify the vibration conditions. The obtained results showed that the ANN-based model outperform the other well-known methods, including Support Vector Machine, Naive Bayes, and decision tree classifiers.

The above literature review reveals that most of the research focused on detections/diagnoses. Not much research has focused on prediction of machine tools vibrations, especially before the operation begins. Furthermore, although the above mentioned studies reveal that ANN-based models have been successfully used in the area of predicting/diagnosing machine fault, however, in order to increase the reliability of prediction/diagnosis results of the ANN-based model, attention is needed to focus on optimizing the parameters of the model. In other words, the training phase plays an important role in developing the ANN-based models. In the literature we examined, the BP algorithm, a gradient-based algorithm, has been widely used in the training phase. However, the BP algorithm has some drawbacks. Heuristic algorithms have been proposed for the purpose of training neural networks to enhance the problems of BP-based algorithms. Recently, the GSA, based on the law of gravity and mass interactions, was developed for solving optimization problems (Rashedi, 2009). Through some benchmarking studies, this algorithm has been proven to be powerful and is considered to outperform other algorithms.

Taking into account the available literature, there is still room for improvement of the ANN models in the problem of predicting/diagnosing machine faults. In this study, we propose an approach based on the ANN improved by the GSA for predicting machine vibrations. Although the model is developed for a specific problem, it can also be used as a basic guide for other problems.

3. ARTIFICIAL NEURAL NETWORKS

An ANN has two types of basic components: neurons and links. A neuron is a processing element and a link is used to connect one neuron with another. Each link has its own weight. Each neuron receives stimulation from other neurons, processes the information, and produces an output. Neurons are organized into a sequence of layers. The first and the last layers are called input and output layers, respectively, and the middle layers are called hidden layers. The input layer presents data to the network. It is not a neural computing layer because it has no input weights and no activation functions. The hidden layer has no connections to the outside world. The output layer presents the output response to a given input. The activation coming into a neuron from other neurons is multiplied by the weights on the links over which it spreads, and is then added together with other incoming activations. A neural network in which activations spread only in a forward direction, from the input layer through one or more hidden layers to the output layer, is known as an FNN. For a given set of data, an FNN can provide a good non-linear relationship. Studies have shown that an FNN even with only one hidden layer can approximate any continuous function (Funahashi, 1989). Therefore, FNN is an attractive approach (Norgaard et al., 2000)]. FNNs have been applied to a wide variety of problems arising from a variety of disciplines, including mathematics, computer science, and engineering (Li et al., 2013). There are two types of FNN, including single-layer perceptron (SLP) and multi-layer perceptron (MLP). An SLP including only a single perceptron is suitable for solving linear problems. Whereas, an MLP with more than one perceptron, organized in different layers, has the capability to solve non-linear problems. Figure 1 shows an example of an MLP with one hidden layer.

In Figure 1, *R*, *N*, and *S* are the numbers of input, hidden neurons, and output, respectively; *iw* and *hw* are the input and hidden weight matrices, respectively; *hb* and *ob* are the bias vectors of the hidden and output layers, respectively; *x* is the input vector of the network; *ho* is the output vector of the hidden layer; and *y* is the output vector of the network. The neural network in Figure 1 can be expressed by the following equations:

$$ho_i = f\left(\sum_{j=1}^{R} iw_{ij}.x_j - hb_i\right), \text{ for } i=1,..,N \tag{1}$$

$$y_i = f\left(\sum_{k=1}^{N} hw_{ik}.ho_k - ob_i\right), \text{ for } i=1,..,S \tag{2}$$

where *f* is an activation function, iw_{ij} is the connection weight from the *j*th node in the input layer to the *i*th node in the hidden layer, and hw_{ik} is the connection weight from the *k*th node in the hidden layer to the *i*th node in the output layer.

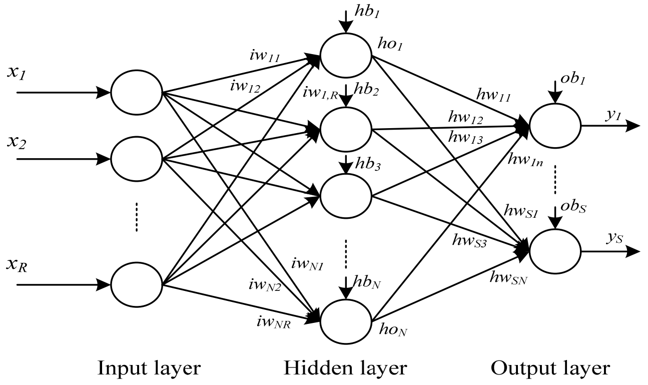

Figure 1. A feed-forward network with three layers.

When implementing a neural network, it is necessary to determine the structure in terms of number of layers and number of neurons in the layers. The larger the number of hidden layers and nodes, the more complex the network will be. A network with a structure that is more complicated than necessary may over fit the training data (Caruana et al., 2001). This means that it performs well on data included in the training set, but may perform poorly on that in a testing set.

Commonly, the design of an artificial neural network consists of three steps as follows: (1) configuration – the way layers are organized and connected, (2) learning – the way information is stored, and (3) generalization – the way neural network produces output when inputs are not in the training set. In this study, a feed-forward neural network combined with supervised learning is proposed to develop ANN-based models.

The most common training algorithms are those derived from back-propagation algorithm (Cruz et al., 2011). The algorithm has two phases, including propagation and weight update. When an input vector is presented to the neural network, it is propagated forward through the network, layer by layer, until it reaches the output layer. The output of the network is then compared to the desired output, and an error value is calculated for each of the neurons in the output layer. The error values are then propagated backwards, starting from the output, until each neuron has an associated error value which represents its contribution to the original output. After training, the neural network is tested to verify its capability of generalizing to new values that are not in the training dataset. Therefore, the neural network works like a black box.

4. GRAVITATIONAL SEARCH ALGORITHM

Optimization problems are concerned with finding the values for one or several decision variables that meet the objectives without violating the constraints. Depending on the objective function, optimization problems might have multiple solutions some of which might be local optima. The goal of the heuristic algorithm is to find as good solution as possible for all instances of the problem. The main idea of all optimization problems solved by heuristic algorithms is that they start off with a more or less arbitrary initial solution, iteratively produce new solutions by some generation rule and evaluate these new solutions, and eventually update the best solution found during the search process (Maringer, 2005). The execution of the iterated search procedure is usually stopped when there has been no further improvement over a given number of iterations (or further improvements cannot be expected); when the found solution is good enough; when the allowed execution time (or other external limit) has been reached; or when some internal parameter terminates the algorithm’s execution. Among recent powerful algorithms, GSA is a heuristic optimization algorithm which has been attracting interest from research community.

The GSA is based on the physical law of gravity and the law of motion. In the universe, every particle attracts every other particle with a gravitational force that is directly proportional to the product of their masses and inversely proportional to the square of the distance between them. The GSA can be considered as a system of agents, called masses, that obey the Newtonian laws of gravitation and masses. All masses attract each other by the gravity forces between them. A heavier mass has a bigger force. The detailed algorithm can be found in Rashedi’s work (2009). The pseudo code of the GSA is given in Figure 2. Commonly, the stopping condition of the algorithm is the number of iterations.

```
begin
Generate an initial population of $N$ agents $X_i$ $(i=1,2,...,N)$
Calculate objective function $f(X)$ for each agent
while (t <MaxGeneration) or (stop criterion)
    Evaluate the fitness for each agent
    for i = 1 to $N$ do
    Update the $G(t)$, $best(t)$, $worst(t)$ and $M_i(t)$ of the population
    end for
    Calculate the total force in different directions.
    Calculate acceleration and velocity
    Update agents’ position
end while
Return the optimum solution in the search space
end
```

Figure 2. Pseudo code of the GSA.

In GSA, the position of the mass corresponds to a solution of the problem, and its gravitational and inertial masses are determined by the use of a fitness function. In other words, each mass presents a solution, and the algorithm is navigated by adjusting the gravitational and inertia masses. It is expected that masses be attracted by the heaviest mass. This mass will present an optimum solution in the search space. Therefore, the outputs of algorithm are masses corresponding to optimal solutions in the search space.

5. PROPOSED HYBRID ALGORITHM FOR TRAINING ANN

The proposed hybrid GSABP algorithm is described in this section. The encoding strategy and the training for FNNs are also presented.

5.1. Improved GSA algorithm

In the improved GSA algorithm, in each iteration, the velocity of agent *i* is updated as follows:

$$v_i(t+1) = w \times v_i(t) + c_1' \times rand \times a_i(t) + c_2' \times rand \times (gbest - x_i(t)), \tag{3}$$

where $v_i(t)$ is the velocity of agent *i* at iteration *t*, c_1' and c_2' are acceleration coefficients, *w* is a weight indicating the effect of previous velocity $v_i(t)$ on the new velocity $v_i(t+1)$ and has a value ranged from 0 to 1, *rand* is a random value in the range of [0,1], $a_i(t)$ is the acceleration of agent *i* at iteration *t*, and *gbest* is the best solution so far. Because *gbest* is the best solution so far, *gbest* may be changed and updated after each calculation. By using (16), the velocities and positions of next agents are calculated based on this information, so that the agents will know their relative positions in comparison with the *gbest* so far. This, in turn, helps agents to move to better positions. The positions of agents are then calculated as follows:

$$X_i(t+1) = X_i(t) + v_i(t+1) \tag{4}$$

To summarize, in the improved GSA, all agents are first randomly generated. Each agent is a candidate solution. The force, gravitational constant, and the total force among agents are then calculated by the use of Equations (4), (9), and (5), respectively. The accelerations of agents are obtained by Equation (6). The best solution is updated in each iteration. The velocities of all agents are calculated by Equation (18). Finally, the positions of agents are achieved by Equation (19).

5.2. Hybrid GSABP algorithm

The objective of the training phase is to find the best combination of the connection weights and biases in the FNN to achieve the minimum error for a training dataset. In a multi-dimensional problem like the FNN architecture, the optimization function has a lot of local minima. For this type of problems, we need an algorithm with ability of escaping from local minima and achieving a near global optimum. The GSA algorithm has a strong ability to search for the global optimum. However, when nearing the global optimum, the search tends to slow down significantly. The BP algorithm has a strong ability to search local optimum, but its ability to search global optimum is weak. The hybrid GSABP is proposed to combine the global search ability of GSA with the local search ability of BP. This combination takes advantage of both algorithms to optimize the weights and biases of the FNN. The fundamental process of the hybrid algorithm is as follows: at the beginning stage, the GSA is employed to search for the optimum, and find near optimal solutions. When the cost function value has not changed for several iterations, the BP algorithm is applied to find the best solution around global optimum. In this way, the hybrid algorithm may find an optimum quickly.

For the BP-based algorithm, the Gradient descent with momentum (GDM) is utilized. The GDM is a special variant of the Gradient algorithm. The parameters updates in GDM are obtained by using the equation:

$$x_{t+1} = x_t - \mu . g_t + \alpha (x_t - x_{t-1}) \tag{5}$$

where x_{t+1} is the new updated vector of weights and biases, x_t is the current vector of weights and biases, μ is a scalar called the learning rate, g_t is the cost function gradient with respect to the vector, and α is the momentum and have the range between 0 and 1. The momentum accelerates the neural network training. The learning rate and momentum used in this study are 0.4 and 0.3, respectively. Figure 3 shows the flowchart of proposed hybrid GSABP algorithm. The procedure for the proposed hybrid algorithm can be described as follows:

Step 1: Initialize an FNN structure and the parameters in the GSA.

Step 2: Set up the encoding relationship between the FNN structure and the GSA parameters.

Step 3: Evaluate the fitness of all agents.

Step 4: Update *G*, *best(t)*, *worst(t)*, and $M_i(t)$ for the population.

Step 5: Calculate forces, and acceleration of all agents.

Step 6: Update agents' position and velocity.

Step 7: If the stop condition is satisfied, go to Step 8; otherwise go to Step 3.

Step 8: Decode the near optimal solution of the FNN structure.

Step 9: Use the BP algorithm to search around the near optimal solution.

Step 10: Return the global optimum.

So far, there is no well-defined rule or procedure to have an optimal network architecture (Khalaj et al., 2014; Nazari, 2013). Therefore, the optimal number of neurons in the hidden layer is identified by using a trial and error approach. The optimum number of neurons in the hidden layer is determined by varying their number, starting with a specific number of neurons, and then increasing in steps by adding one neuron each time. Hence, various network architectures were tested to achieve the optimum number of hidden neurons. The best performing FNN architecture was then identified, and thus providing the results with the smallest error values during the training. It is noted that the number of neurons in the hidden layer is restricted to integer values. The parameters, including weights and biases, are allowed to accept the real values. These parameters are optimized by the hybrid algorithm.

5.3. The fitness function in training neural networks

Figure 1 shows an FNN with one input, one hidden, and one output layer. Suppose that the activation function from input to hidden is a sigmoid function, and the activation function from hidden to output is a linear function. The fitness function is given as follows:

The output of each hidden node is calculated as:

$$f(s_j) = 1/(1 + \exp(-s_j)), \; j=1,2,...,N, \tag{6}$$

where $s_j = \sum_{i=1}^{R} iw_{ji}.x_i - hb_j$, *R* is the number of input nodes, iw_{ji} is the connection weight from *i*th node of the input layer to the *j*th node of the hidden layer, hb_j is the bias of the *j*th hidden node, x_i is the *i*th input. The *k*th output node is:

$$y_k = \sum_{j=1}^{N} hw_{kj}.f(s_j) - ob_k \, , \; k=1,2,...,S, \tag{7}$$

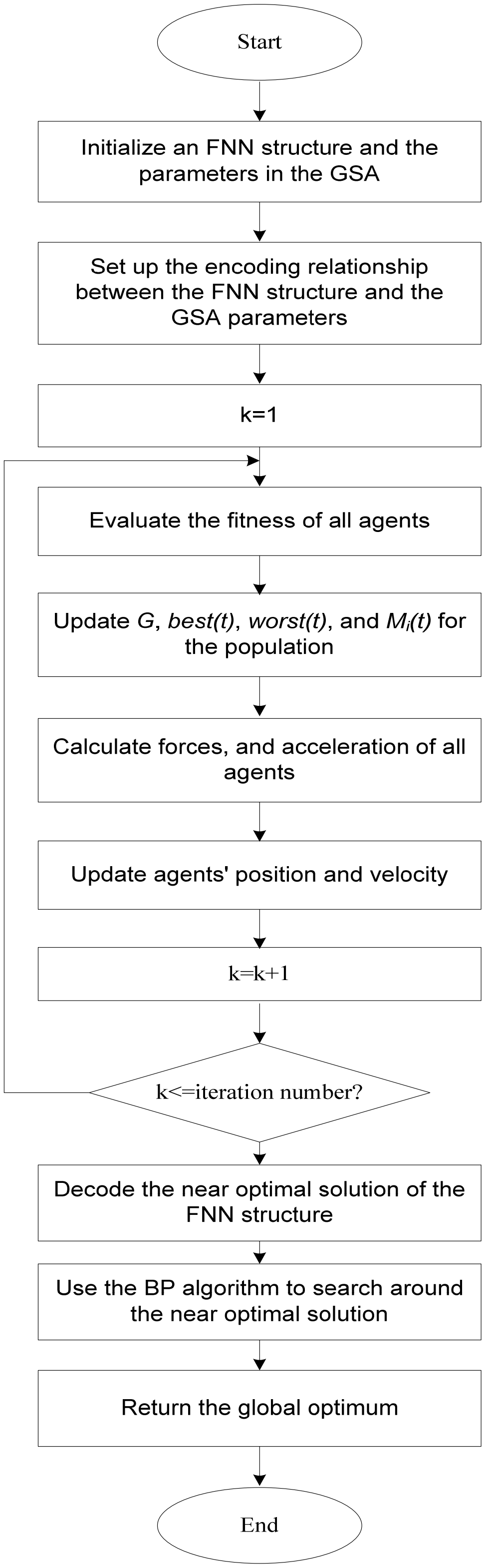
Start
Initialize an FNN structure and the parameters in the GSA
Set up the encoding relationship between the FNN structure and the GSA parameters
k=1
Evaluate the fitness of all agents
Update G, best(t), worst(t), and Mi(t) for the population
Calculate forces, and acceleration of all agents
Update agents' position and velocity
k=k+1
k<=iteration number?
Decode the near optimal solution of the FNN structure
Use the BP algorithm to search around the near optimal solution
Return the global optimum
End

Figure 3. The vector of training parameters

where hw_{kj} is the connection weight from the *j*th hidden node to the *k*th output node, ob_k is the bias of the *k*th output node.

The learning error E (the fitness function) can be calculated as:

$$E = \sum_{k=1}^{Q} E_k / Q \,, \quad E_k = \sum_{i=1}^{S} \left(y_i^k - d_i^k \right)^2 \,, \tag{8}$$

where Q is the number of training samples, y_i^k is the actual output of the ith input when the kth training sample is used, and d_i^k is the desired output of the ith input when the kth training sample is used. The fitness function of the kth training sample is:

$$fitness\left(X_k\right) = E_k \,. \tag{9}$$

5.4. Encoding strategy

There are three ways to encode and represent the weights and biases of FNN for every solution in evolutionary algorithms (Zhang et al., 2013). They are the vector, matrix, and binary encoding methods. In this study, we utilized the vector encoding method. The objective function is to minimize SSE and the proposed hybrid GSABP algorithm was used to search optimal weights and biases of neural networks. The amount of errors is determined by the squared difference between the target output and actual output. As shown in Figure 4, in the implementation of the hybrid algorithm to train a neural network (GSABP-FNN), all training parameters, *θ={iw, hw, hb, ob}*, are converted into a single vector of real numbers and each vector represents the weights and biases of a FNN.

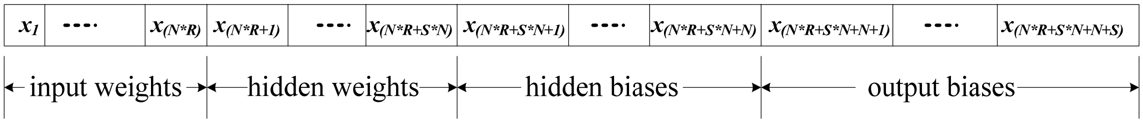

Figure 4. The vector of training parameters

6. EXPERIMENTAL DESIGN

In general, ANN training can be considered as reforming parameters, including weights and biases, that are randomly initialized. These parameters are modified in each iteration until the error between ANN output and actual output corresponding to the input is as small as possible. So, updating the ANN parameters leads to an optimization problem. In this study, the hybrid GSABP algorithm is used to adjust the parameters of ANNs. Updating the neural network weights using the GSABP algorithm considering the training neural network as a function and the goal must be the optimization of this function in an n-dimensional space.

Section 6 illustrates an application of the proposed ANN-based model to the prediction of vibrations. The CNC milling machine is used as an illustration. In order to simplify the experiment, we focus on plane-surface machining. Furthermore, it is assumed that both the machine tool and the cutting tool are in good conditions, the cutting tool (end mill cutter, S45C) is suitable for cutting the work-piece (cast iron), the work-piece is properly held, only up milling is considered, and no cutting fluid is used. (Note that without the above assumptions, the dataset would contain very huge data). During the experiment, a vibration analyzer, VA12, was used to measure the vibration. The vibration analyzer can measure vibrations based on the acceleration, velocity, and displacement. In this study, the vibration was measured according to the velocity. The dataset was collected by an expert working with a machining factory in Taiwan. The data collection steps include (1) putting a set of input parameters into the NC panel, (2) using the vibration analysis instrument (VA-12) to measure the data of velocity, (3) measuring the classification of vibration class according to the velocity. The datasets and programs can be requested by contacting the author by email.

6.1 CNC milling machines

CNC milling machine is a special form of computer numerical control (CNC) machine. In general, CNC milling machines are grouped by the number of axes on which they operate, which are labeled with various letters. X and Y designate horizontal movement of the work-piece (forward-and-backward and side-to-side on a flat plane). Z represents vertical movement. Although CNC milling machines are ideal solutions from prototyping and short-run production of complex parts to the manufacturing of distinctive precision parts, a high-quality milling machine requires a stiff frame in order to maintain precise positioning when encountering strong forces at high cutting speeds.

6.2 Structure of CNC milling machines

The structure of a milling machine includes two sectors: one consists of base, saddle, table, head stock and column (Figure 5), and the other consists of spindle, magazine, NC panel, cover, and so on, as illustrated in Figure 6.

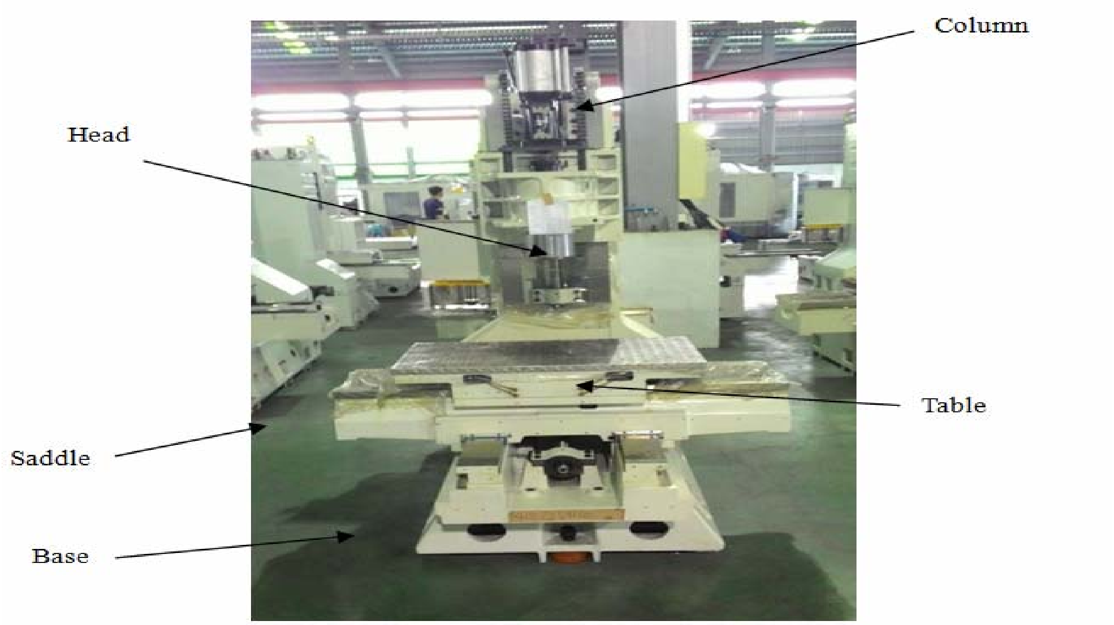

Figure 5. The cast parts of a milling machine.

6.3. Dynamic behavior and vibration of a milling machine

Machinery systems can encounter free vibration, forced vibration and self-excited vibration (chatter) during machining. In general, free vibration and forced vibration are less destructive compared to chatter. The self-excited vibration is induced by forces generated in the cutting process (Siddhpura and Paurobally, 2012). For example, increasing spindle speeds may result in the onset of chatter. Chatter can create large cutting forces and thus may accelerate tool wear and cause tool failure, leading to the vibrations of machine tools.

In order to analyze chatter’s dynamic behavior, the rigidity and stability are the two most important characteristics that need to be taken into account. The conditions of rigidity and stability of the machine tools can cause vibrations during machining processes. The rigidity or stability changes resulted from several phenomena including (1) chip thickness variation; (2) penetration rate variation; and (3) cutting speed variation. Each of these three phenomena in turn is caused by several input parameters during operations (see Figure 7). For example, the chip thickness variation is affected by (a) the number of teeth (z_n); (b) the cutter diameter (D_c); and (c) the depth of cut/axial depth of cut (a_p,). The penetration rate variation is affected by (a) the feed per tooth (f_z); and (b) the feed speed (v_f). The cutting speed variation is affected by (a) the cutting speed (v_c); and (b) the revolutions per minute (*m-rpm*).

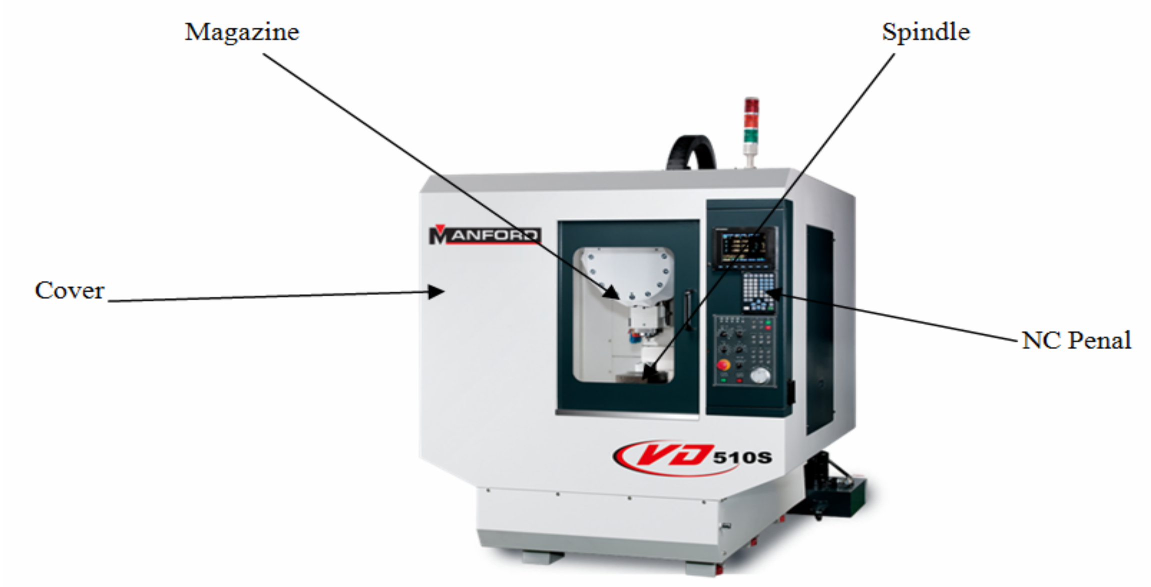

Figure 6. The accessory parts of a milling machine.

6.4. Identifying input and output variables

Based on the discussion above and through literature review, operation parameters (1) the number of teeth (z_n), (2) the cutter diameter (D_c), (3) the depth of cut/axial depth of cut (a_p,), (4) the feed per tooth (f_z), (5) the feed speed (v_f), (6) the cutting speed (v_c), and (7) the revolutions per minute (*m-rpm*) may have direct or indirect influence on the tools' or work-pieces' vibration. Therefore, in this research we treat these parameters as input variables and the output variable (*Y*) is the degree of vibration.

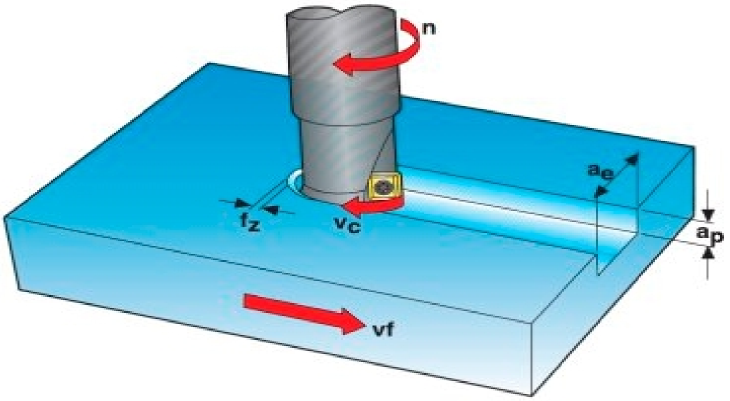

Figure 7. An example of a milling operation.

6.5. Dataset

We obtained our data from a machining factory in Taiwan. The dataset consists of 480 cases and was divided into two groups: the first group (about 60%) was used for training the model and the second group (about 40%) was employed for testing the model. The training dataset served in model building while the other group was used for the validation of the developed model.

6.6. Examining the performance

To examine the performance of a neural network, several criteria can be used. These criteria are applied to the trained neural network to know how well it works. The criteria used to compare predicted values and actual values are as follows:

Root mean squared error (RMSE): This index estimates the residual between the actual value and predicted value. A model has better performance if it has a smaller RMSE. An RMSE equal to zero represents a perfect fit. The RMSE is calculated as:

$$RMSE = \sqrt{\frac{1}{m}\sum_{k=1}^{m}\left(t_k - y_k\right)^2}\ , \tag{10}$$

where t_k is the actual value, y_k is the predicted value produced by the model, and m is the total number of observations.

Mean absolute percentage error (MAPE): This index indicates an average of the absolute percentage errors; the lower the MAPE the better it is. The MAPE is calculated as:

$$MAPE = \frac{1}{m}\sum_{k=1}^{m}\left|\frac{t_k - y_k}{t_k}\right|. \tag{11}$$

Mean absolute error (MAE): This index indicates how close the predicted values are to the actual values and is calculated as:

$$MAE = \frac{1}{m}\sum_{k=1}^{m}\left|t_k - y_k\right| \tag{12}$$

Correlation coefficient (R): This criterion reveals the strength of relationships between actual values and predicted values. The correlation coefficient has a range from -1 to 1, and a model with a higher R indicates it has better performance. It is calculated as:

$$R = \frac{\sum_{k=1}^{m}\left(t_k - \bar{t}\right)\left(y_k - \bar{y}\right)}{\sqrt{\sum_{k=1}^{m}\left(t_k - \bar{t}\right)^2 . \sum_{k=1}^{m}\left(y_k - \bar{y}\right)^2}}\ , \tag{13}$$

where $\bar{t}$ and $\bar{y}$ are the average values of t_k and y_k, respectively.

In addition to the aforementioned criteria, the number of iterations required by individual training algorithms to reach the certain output accuracy was also used to evaluate the performance of the training algorithms.

7. EXPERIMENTAL RESULTS AND DISCUSSION

The models were coded and implemented in the Matlab environment (Matlab R2014a). As discussed earlier, one hidden layer was used. The optimum number of neurons in the hidden layer was determined by varying their number, starting with a minimum of one, and then increasing in steps by adding one neuron each time. Hence, various network architectures were tested to achieve the optimum number of hidden neurons. The best performing architecture was 7-15-1, i.e., with one hidden layer and 15 neurons, resulting in a total of 120 weights and 16 biases.

In order to evaluate the proposed algorithm, we used BP, GSA, and GSABP algorithms to search optimal weights and biases of FNNs; hereafter, we refer to them as FNN-BP, FNN-GSA and FNN-GSABP. Suppose that weights and biases were initially set in the range of [-10, 10]. For FNN-GSA, α was set to 20, the initial velocities of particles were randomly generated in the range of [0, 1], the initial values of acceleration and mass were set to 0 for each particle, and the gravitational constant (G_0) was set to 100. For FNN-BP, the learning and momentum rates were to 0.4 and 0.3, respectively. For FNN-GSABP, α was set to 20, the initial velocities of particles were randomly generated in the range of [0, 1], the initial values of acceleration and mass were set to 0 for each particle, the gravitational constant (G_0) was set to 100, w was set to 0.6, c_1' was set to 0.5, c_2' was set to 1.5, and the learning and momentum rates were set to 0.4 and 0.3, respectively. The population sizes of FNN-GSA and FNN-GSABP were both set at 50.

In this research, the number of iterations was chosen as the stopping criterion. Figure 8 depicts the RMSE values obtained in the training phase for the three models in 1000 iterations. At the 1000th iteration, the RMSE values of the FNN-BP, FNN-GSA, and FNN-GSABP were 0.0557, 0.0439, and 0.0303, respectively. The FNN-PB and FNN-GSABP had a faster convergence than the FNN-GSA. However, The FNN-BP was trapped in local minima of the parameter space, and therefore yielded a poor performance. Among the three models, the FNN-GSABP has the capability of avoiding premature convergence and exploring the whole search space.

Table 1 gives the performance statistics of the FNN-BP, FNN-GSA and FNN-GSABP at the 1000*th* iteration. Theoretically, a prediction model is accepted as ideal when MAPE, RMSE, and MAE are small and *R* is close to 1. At the 1000th iteration, the performance statistics MAPE, RMSE, MAE, and R obtained on the testing dataset by the FNN-GSABP model were calculated as 0.2297, 0.0503, 0.0318, and 0.9378, respectively. These results were highly correlated. The FNN-GSABP has smaller MAPE, RMSE, and MAPE values as well as bigger *R* values for both the training and testing datasets. This means that the FNN-GSABP has a better overall performance in all criteria.

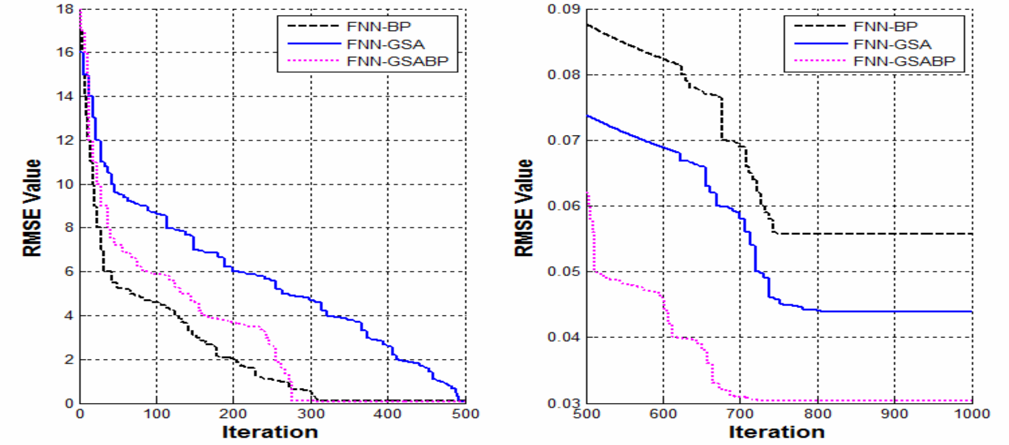

Figure 8. RMSE values in 1000 iterations.

Table 1: Performance statistics of FNN-BP, FNN-GSA, and FNN-GSABP

Model	Training dataset				Testing dataset			
	MAPE	RMSE	MAE	R	MAPE	RMSE	MAE	R
FNN-BP	0.3178	0.0557	0.0377	0.8331	0.4317	0.0788	0.0540	0.8411
FNN-GSA	0.2333	0.0439	0.0292	0.8957	0.3234	0.0740	0.0448	0.8724
FNN-GSABP	0.1797	0.0303	0.0211	0.9390	0.2297	0.0503	0.0318	0.9378

Figure 9 presents the scatter diagrams that illustrate the degree of correlation between actual values and predicted values obtained by the FNN-GSABP after 1000 iterations. In the figure, an identity line was drawn as a reference of perfect agreement. In this problem, the identity line means that the predicted values and actual values are identical. The more the two values agree, the more the points tend to concentrate in the vicinity of the identity line. It may be observed that most predicted values are close to the actual values. This trend indicates a good agreement between these two models' predictions and the actual values. Based on the results, it can be concluded that the FNN-GSABP model can be used to predict machine vibrations. Regarding result accuracy, the FNN- GSABP is highly appreciated. When given enough computation, the hybrid algorithm GSABP can find the optimal solution. Therefore, the FNN-GSABP outperformed the FNN-GSABP and FNN-BP, and the results show that its prediction is more accurate and reliable. Hence, the FNN-GSABP may be acceptable to serve as a predictor of machine vibrations.

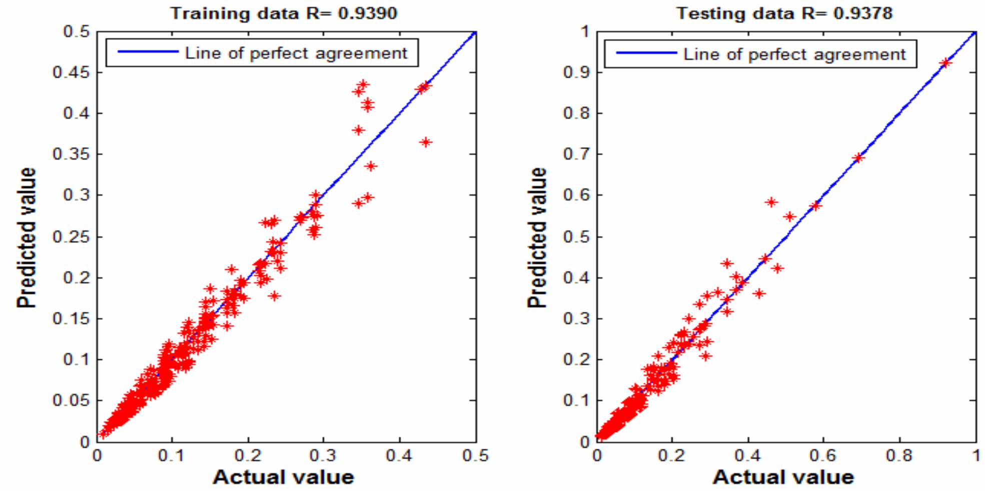

Figure 9. Comparison between actual and predicted values for the FNN-GSABP model at the 1000*th* iteration.

8. CONCLUSIONS

In the research, the FNN was utilized to predict the vibrations. We presented a hybrid of improved GSA and GDM algorithms to train neural networks for the prediction of machine vibrations. The performance statistics of the FNN-GSABP were compared against those of the FNN-GSA and FNN-BP in terms of MAPE, RMSE, MAE, and R achieved. The FNN-GSABP was found to have better overall performance in all criteria. The findings demonstrated the remarkable advantage of the proposed hybrid algorithm and the potential application of ANN in the prediction/diagnosis machine fault area. It is expected that this work may be used as a supportive tool to assist staff in machine tool industry to predict/diagnose machine vibrations.

REFERENCES

Adeli, H. and Hung, S.L. 1994. An adaptive conjugate gradient learning algorithm for efficient training of neural networks, *Applied Mathematics and Computation* **62**(1):81-102.

Aghajanloo, M.B., Sabziparvar, A.A. and Talaee, P.H. 2013. Artificial neural network–genetic algorithm for estimation of crop evapotranspiration in a semi-arid region of Iran, *Neural Computing and Applications* **23**(5):1387-1393.

Azar, D., Fayad, K., Daoud, C. 2016. A combined ant colony optimization and simulated annealing algorithm to assess stability and fault-proneness of classes based on internal software quality attributes, *International Journal of Artificial Intelligence* **14**(2):137-156.

Barakat, M., Lefebvre, D., Khalil, M., Druaux, F. and Mustapha, O. 2013. Parameter selection algorithm with self adaptive growing neural network classifier for diagnosis issues, *International Journal of Machine Learning and Cybernetics* **4**(3):217-233.

Caruana, R., Lawrence, S. and Giles, C.L. 2001. Overfitting in neural networks: backpropagation, conjugate gradient, and early stopping, Proceedings of 13th Conference on Advances Neural Information Processing Systems, USA, 402-408.

Chen, J.-F.; Lo, S.-K.; Do, Q.H. 2016. An Approach to the Classification of Cutting Vibration on Machine Tools. *Information* 7(7).

Chen, P. 2013. Research on Engineering Machinery Fault Diagnosis based on Neural Network, *Journal of Theoretical and Applied Information Technology* **49**(2):771-777.

Duman, S., Güvenç, U. and Yörükeren, N. 2010. Gravitational Search Algorithm for Economic Dispatch with Valve-Point Effects, *International Review of Electrical Engineering* **5**(6): 2890-2895.

Funahashi, K. 1989. On the approximate realization of continuous mappings by neural networks, *Neural Networks* **2**(3):183-192.

Gupta, J.N.D. and Sexton, R.S. 1999. Comparing backpropagation with a genetic algorithm for neural network training, *Omega* **27**:679-684.

Hagar, M.T. and Menhaj, M.B. 1994. Training feedforward networks with the Marquardt algorithm, *IEEE Transactions on Neural Networks* **5**(6):989–993.

Hush, D.R. and Horne, N.G. 1993. Progress in supervised neural networks, *IEEE Signal Processing Magazine* **10**(1):8-39.

Jack, L.B. and Nandi, A.K. 2002. Fault Detection using Support Vector Machines and Artificial Neural Networks, Augmented by Genetic Algorithms, *Mechanical Systems and Signal Processing* **16**(2-3):373-390.

Kalkat, M. 2014. Experimentally vibration and noise analysis of two types of washing machines with a proposed neural network predictor, *Measurement* **47**:184–192.

Khalaj, G., Nazari, A., Yoozbashizadeh, H., Khodabandeh, A. and Jahazi, M. 2014. ANN model to predict the effects of composition and heat treatment parameters on transformation start temperature of microalloyed steels, *Neural Computing and Applications* **24**(2):301-308.

Khodja, D.J. and Chetate, B. 2005. Development of Neural Networks module for fault identification in asynchronous machine using various types of reference signals, *Proceedings of International Conference on Physics and Control*, 531-536.

Kumar, K.P., S. Rao, K.V.N. and Krishna, K.R. 2012. Neural network based vibration analysis with novelty in data detection for a large steam, *Journal of Shock and Vibration* **19**(1):25-35.

Lei, Y., He, Z., Zi, Y. and Hu, Q. 2008. Fault diagnosis of rotating machinery based on a new hybrid clustering algorithm, *International Journal of Advanced Manufacturing Technology* **35**: 968-977.

Liao, X., Wan, Z., Li, Y. and Cheng L. 2013. Fault Diagnosis Method of Rolling Bearing Based on Ensemble Local Mean Decomposition and Neural Network, *Applied Mechanics and Materials* **300-301**:714-720.

Li, L.K., Shao, S. and Yiu, K.F.C. 2013. A new optimization algorithm for single hidden layer feedforward neural networks, *Applied Soft Computing* **13**(5):2857-2862.

Lin, C.C. 2011. Intelligent Vibration Signal Diagnostic System Using Artificial Neural Network, Artificial Neural Networks, *Industrial and Control Engineering Applications*, InTech Publisher, 421-440.

Maringer, D.G. 2005. Portfolio Management with Heuristic Optimization, *Advances in Computational Management Science*, **8**:38-76.

Martín, D., Caballero, B., Haber, R. 2009. Optimal tuning of a networked linear controller using a multi-objective genetic algorithm and its application to one complex electromechanical process, *International Journal of Innovative Computing, Information and Control* B(10B):3405-3414.

Mirjalili, S.A., Hashim, S.Z.M. and Sardroudi, H.M. 2012. Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm, *Applied Mathematics and Computation* **218**(22):11125-11137.

Moosavian, A., Ahmadi, H., Tabatabaeefar, A. and Khazaee, M. 2013. Comparison of two classifiers; K -nearest neighbor and artificial neural network, for fault diagnosis on a main engine journal-bearing, *Shock and Vibration* 20:263-272.

Nazari, A. 2013. Artificial neural networks for prediction compressive strength of geopolymers with seeded waste ashes, *Neural Computing and Applications* **23**:391-402.

Norgaard, M.R.O., Poulsen, N.K. and Hansen, L.K. 2000. Neural networks for modeling and control of dynamic systems. A practitioner's handbook, Springer, London.

Nyanteh, Y. and Srivastava, S. 2013. Application of Artificial Intelligence to Real-Time Fault Detection in Permanent-Magnet Synchronous Machines, *IEEE Transactions on Industry Applications* **49**(3):1205-1214.

Precup, R.E., David, R.C., Petriu, Emil M., Preitl, S., Radac, M.B. 2014. Novel adaptive charged system search algorithm for optimal tuning of fuzzy controllers, *Expert Systems with Applications* **41**(4):1168-1175.

Prieto, M.D., Cirrincione, G., Espinosa, A.G., Ortega, J.A. and Henao, H. 2013. Bearing Fault Detection by a Novel Condition-Monitoring Scheme Based on Statistical-Time Features and Neural Networks, *IEEE Transactions on Industrial Electronics* **30**(8):3398-3407.

Ranković, V., Novaković, A., Grujović, N., Divac D. and Milivojević, N. 2014. Predicting piezometric water level in dams via artificial neural networks, *Neural Computing and Applications* **24**(5): 1115-1121.

Rashedi, E., Nezamabadi-pour, H. and Saryazdi S. 2009. GSA: A Gravitational Search Algorithm, *Information Sciences* **179**(13):2232-2248.

Rossana M. S. Cruz, Helton M. Peixoto and Rafael M. Magalhães 2011. *Artificial Neural Networks and Efficient Optimization Techniques for Applications in Engineering, Artificial Neural Networks - Methodological Advances and Biomedical Applications*, InTech.

Siddhpura, M., Paurobally, R. 2012. A review of chatter vibration research in turning. I*nternational Journal of Machine Tools and Manufacture* **61**:27-47.

Solos, I. P., Tassopoulos, I. X., Beligiannis, G.N. 2016. Optimizing shift scheduling for tank trucks using an effective stochastic variable neighbourhood approach, *International Journal of Artificial Intelligence* **14**(1):1-26.

Su, H., Chong, K.T. and Parlos, A.G. 2005. A Neural Network Method for Induction Machine Fault Detection with Vibration Signal. Computational Science and Its Applications – ICCSA 2005, *Lecture Notes in Computer Science* **3481**:1293-1302.

Taghavifar, H., Mardani A. 2014. Use of artificial neural networks for estimation of agricultural wheel traction force in soil bin, *Neural Computing and Applications* **24**(6):1249-1258.

Ye, F., Zhang, Z., Chakrabarty, K. and Gu, X. 2013. Board-Level Functional Fault Diagnosis Using Artificial Neural Networks, Support-Vector Machines, and Weighted-Majority Voting, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **32**(5):723-736.

Zhang, N. 2009. An online gradient method with momentum for two-layer feedforward neural networks, *Applied Mathematics and Computation* **212**(2):488-498.

Zhang, J.R., Zhang, J., Lock, T.M. and Lyu, M.R. 2007. A hybrid particle swarm optimization–back-propagation algorithm for feedforward neural network training, *Applied Mathematics and Computation* **185**(2):1026-1037.

Zhang, T., Wang, Z., Yu, T., Wang, W. and Zhao, H. 2013. Research on fault diagnosis for TBM Based on wavelet packet transforms and BP neural network, *Proceedings of the IEEE 3rd International Advance Computing Conference* (IACC), 677-681.