Automation in Construction 16 (2007) 642 – 646

AUTOMATION IN CONSTRUCTION

www.elsevier.com/locate/autcon

Application of a PSO-based neural network in analysis of outcomes of construction claims

K.W. Chau

Department of Civil and Structural Engineering, Hong Kong Polytechnic University, Hunghom, Kowloon, Hong Kong

Accepted 9 November 2006

Abstract

It is generally acknowledged that construction claims are highly complicated and are interrelated with a multitude of factors. It will be advantageous if the parties to a dispute have some insights to some degree of certainty how the case would be resolved prior to the litigation process. By its nature, the use of artificial neural networks (ANN) can be a cost-effective technique to help to predict the outcome of construction claims, provided with characteristics of cases and the corresponding past court decisions. This paper presents the adoption of a particle swarm optimization (PSO) model to train perceptrons in predicting the outcome of construction claims in Hong Kong. It is illustrated that the successful prediction rate of PSO-based network is up to 80%. Moreover, it is capable of producing faster and more accurate results than its counterparts of a benchmarking back-propagation ANN. This will furnish an alternative in assessing whether or not to take the case to litigation.

Keywords: Particle swarm optimization; Artificial neural networks; Construction claims

1. Introduction

It is generally recognized that, owing to highly complicated nature of pertinent activities and interrelation with a multitude of factors, the construction industry is particularly vulnerable to litigation. The disagreement between the involved parties can arise from interpretation of the contract, unforeseen site conditions, variation orders by the client, acceleration and suspension of works, and so on. The main forums for the resolution of construction disputes are mediation, arbitration, and the courts. However, the consequence of any disagreements between the client and the contractor may be far reaching. It should be noted that the litigation process, often with the involvement of specialized and complex issues, is usually very expensive. It might be to the best interest of all the involved parties to minimize or even avoid the likelihood of litigation by taking a conscientious management procedure and concerted effort. In this way, they may avoid the often inefficient use of resources, higher costs for both parties through settlement, the damage to the reputation of both sides, and so forth.

It will be advantageous if the parties to a dispute have some insights to some degree of certainty how the case would be resolved prior to the litigation process. Recent artificial intelligence techniques can be used to identify the hidden relationships among various interrelated factors and to predict decisions that will be made by the court, based on characteristics of cases and the corresponding past court decisions. A precise prediction of possible litigation outcomes would effectively help to significantly reduce the number of disputes that would need to be settled by the much more expensive litigation process. Among others, the use of artificial neural networks (ANN) can be a cost-effective technique to help predict the outcome of construction claims. A comprehensive literature review of other research works published on applying ANN for claim resolution predictions (or the like) have been undertaken. However, it is found that AI techniques are not common and are rarely applied in legal field. Arditi et al. [1] is the only one being found.

The ANN, and in particular, the feed forward backward propagation perceptrons, are widely applied in different fields during the past decade [1-9]. It appears that the multi-layer perceptrons can be trained to approximate and accurately generalize virtually any smooth, measurable function whilst

E-mail address: cekwchau@polyu.edu.hk.

0926-5805/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.autcon.2006.11.008

taking no prior assumptions concerning the data distribution. Characteristics, including built-in dynamism in forecasting, data-error tolerance, and lack of requirements of any exogenous input, render it attractive for use in various types of prediction. Although the back propagation (BP) algorithm is commonly used in recent years to perform the training task, some drawbacks are often encountered in the use of this gradient-based method. They include: the training convergence speed is very slow; it is easily to get stuck in a local minimum. Different algorithms have been proposed in order to resolve these drawbacks [10,11,3].

Moreover, Kennedy et al. [12] has performed optimization of feed forward neural nets' structure as well as weights using particle swarms. The use of a particle swarm optimization (PSO) algorithm might furnish an alternative in training the perceptrons of the ANN. This paper presents a PSO-based neural network approach for prediction of the outcome of construction litigation, based on court decisions in the last 10 years in Hong Kong.

A key contribution of the presented research and the unique works done by the author is the adoption of the PSO-based AI techniques tailoring for the prediction of construction litigation outcomes, which is a field where new technological aids are rarely applied. This can be evidenced by the extremely low applications of the recently popular AI techniques to this domain area. A major modification made in this study on the traditional PSO algorithm in order to be applicable to the case application is the adoption of binary coding representation of the domain knowledge, which will be described in more details in the later section. In fact, PSO algorithm is adopted in this application owing to the nature of the domain problem. Special characteristics of the case application that make PSO more suitable than traditional BP include the sufficient amount of the data during the 10 years and the subtle inter-relationships among various principal parameters in determining the outcomes of construction litigation.

2. Disputes in construction

As evidenced by the fact that every site is unique and is never the same as others, the nature of construction activities is varying and dynamic. Thus the preparation of the construction contract can be recognized as the formulation of risk allocation amongst the involving parties: the client, the contractor, and the engineer. The risks involved include unforeseen ground conditions, site instructions, variation orders, the time of completion, the final cost, the quality of the works, client-initiated changes, engineer-initiated changes, errors and omissions in drawings, mistakes in specifications, inflation, inclement weather, delayed payment, changes in regulations, third-party interference, professional negligence, shortage of materials, shortage of plants, labor problems, defects in works, accidents, supplier delivery failure, delay of schedule by subcontractor, poor workmanship, and so forth.

The usual practice is that the involving parties will attempt to sort out the conditions of claims and disputes in the contract documents, well before the actual construction commences. However, since a project usually involves thousands of separate pieces of work items to be integrated together to constitute a complete functioning structure, the potential for honest misunderstanding is extremely high. In Hong Kong, the current setting of disputes resolution is such that the processes of mediation, arbitration, and the courts should be followed successively [13].

3. Artificial neural networks (ANNs)

In a typical multi-layer feed forward perceptron ANN, there exists a nonlinear mapping between input vector and output vector via a system of simple interconnected neurons. It is fully connected to every node in the next and previous layer. The output of a neuron is scaled by the connecting weight and fed forward to become an input through a nonlinear activation function to the neurons in the next layer of network. In the course of training, the perceptron is repeatedly presented with the training data. The weights in the network are then adjusted until the errors between the target and the predicted outputs are small enough, or a pre-determined number of epochs is passed. The perceptron is then validated by presenting with an input vector not belonging to the training pairs. The training processes of ANN are usually complex and high dimensional problems. A fatal drawback of the commonly used gradient-based BP algorithm, which is a local search method, is its easy entrapment into local optimum point.

4. Particle swarm optimization (PSO) algorithm

Particle swarm optimization (PSO) algorithm, which is tailored for optimizing difficult numerical functions and based on metaphor of human social interaction, is capable of mimicking the ability of human societies to process knowledge [14,15]. It has roots in two main component methodologies: artificial life (such as bird flocking, fish schooling and swarming); and, evolutionary computation. Although the PSO algorithm is initially developed as a tool for modeling social behavior, it has been applied in different areas [12,16-20]. Moreover, it has been recognized as a computational intelligence technique intimately related to evolutionary algorithms. Details of the original PSO algorithm can be found in Kennedy et al. [15].

PSO is a populated search method for optimization of continuous nonlinear functions resembling the movement of organisms in a bird flock or fish school. Its key concept is that potential solutions are flown through hyperspace and are accelerated towards better or more optimum solutions. Its paradigm can be implemented in simple form of computer codes and is computationally inexpensive in terms of both memory requirements and speed. It lies somewhere between evolutionary programming and genetic algorithms.

As in evolutionary computation paradigms, the concept of fitness is employed and candidate solutions to the problem are termed particles or sometimes individuals, each of which adjusts its flying based on the flying experiences of both itself and its companions. It keeps track of its coordinates in hyperspace which are associated with its previous best fitness solution, and also of its counterpart corresponding to the overall best value acquired thus far by any other particle in the population. Vectors are taken as presentation of particles since most optimization problems are convenient for such variable presentations.

In fact, the fundamental principles of swarm intelligence are adaptability, diverse response, proximity, quality, and stability. It is adaptive corresponding to the change of the best group value. The allocation of responses between the individual and group values ensures a diversity of response. The higher dimensional space calculations of the PSO concept are performed over a series of time steps. The population is responding to the quality factors of the previous best individual values and the previous best group values. The principle of stability is adhered to since the population changes its state if and only if the best group value changes.

A similarity between PSO and a genetic algorithm is the initialization of the system with a population of random solutions. Instead of employing genetic operators, the evolution of generations of a population of these individuals in such a system is by cooperation and competition among the individuals themselves. Moreover, a randomized velocity is assigned to each potential solution or particle so that it is flown through hyperspace. Whilst he stochastic factors allow thorough search of spaces between regions that are spotted to be relatively good, the momentum effect of modifications of the existing velocities leads to exploration of potential regions of the problem domain. In this way, the adjustment by the particle swarm optimizer is ideally similar to the crossover operation in genetic algorithms whilst the stochastic processes are close to evolutionary programming.

Since the stochastic PSO algorithm has been found to be able to find the global optimum with a large probability and high convergence rate [14,12] , it is adopted to train the multi-layer perceptrons in this case study.

5. Adaptation of PSO to network training

In this application case, a three-layered preceptron is considered. The use of two weight matrices for different layers might appear awkward. In fact, the code is simply set up to work with layers separately and the particle swarm treats the entire set of matrices as one long vector. In the training of the multi-layer preceptrons by the PSO, the representation of the connection weight matrix of the i-th particle is as follows:

$$\mathbf{W}_i = \left\{\mathbf{W}_i^{[1]}, \mathbf{W}_i^{[2]}\right\} \tag{1}$$

where $\mathbf{W}_i^{[1]}$ and $\mathbf{W}_i^{[2]}$ represent the connection weight matrix of the i-th particle between the input layer and the hidden layer, and that between the hidden layer and the output layer, respectively. Moreover, the vector of the position of the previous best fitness value of any particle is represented by

$$\mathbf{P}_i = \left\{\mathbf{P}_i^{[1]}, \mathbf{P}_i^{[2]}\right\} \tag{2}$$

where $\mathbf{P}_i^{[1]}$ and $\mathbf{P}_{\rm i}^{[2]}$ represent the position of the previous best fitness value of the i-th particle, between the input layer and the hidden layer, and that between the hidden layer and the output layer, respectively.

The index of the best particle among all the particles in the population is represented by the symbol b. So the best matrix is represented by

$$\mathbf{P}_b = \left\{\mathbf{P}_b^{[1]}, \mathbf{P}_b^{[2]}\right\} \tag{3}$$

where $\mathbf{P}_b^{[1]}$ and $\mathbf{P}_b^{[2]}$ represent the position of the best particle among all the particles, between the input layer and the hidden layer, and that between the hidden layer and the output layer, respectively.

The velocity of the particle i is denoted by

$$\mathbf{V}_i = \left\{\mathbf{V}_i^{[1]}, \mathbf{V}_i^{[2]}\right\} \tag{4}$$

If m and n represent the index of matrix row and column, respectively, the manipulation of the particles are as follows

$$\begin{aligned}\mathbf{V}_i^{[j]}(m,n) = \mathbf{V}_i^{[j]}(m,n) + r\alpha\left[\mathbf{P}_i^{[j]}(m,n) - \mathbf{W}_i^{[j]}(m,n)\right] \\ + s\beta\left[\mathbf{P}_b^{[j]}(m,n) - \mathbf{W}_i^{[j]}(m,n)\right]\end{aligned} \tag{5}$$

and

$$\mathbf{W}_i^{[j]} = \mathbf{W}_i^{[j]} + \mathbf{V}_i^{[j]} \tag{6}$$

where $j=1, 2$; $m=1, \ldots, M_{\rm j}$; $n=1, \ldots, N_{\rm j}$; $M_{\rm j}$ and $N_{\rm j}$ are the row and column sizes of the matrices **W**, **P**, and **V**; r and s are positive constants; α and β are random numbers in the range from 0 to 1. Eq. (5) is employed to compute the new velocity of the particle based on its previous velocity and the distances of its current position from the best experiences both in its own and as a group. In the context of social behavior, the cognition part $r\alpha[\mathbf{P}_i^{[j]}(m,n) - \mathbf{W}_i^{[j]}(m,n)]$ represents the private thinking of the particle itself whilst the social part $s\beta[\mathbf{P}_b^{[j]}(m,n) - \mathbf{W}_i^{[j]}(m,n)]$ denotes the collaboration among the particles as a group. Eq. (6) then determines the new position according to the new velocity [12,16,20].

The fitness of the i-th particle is expressed in term of an output mean squared error of the neural networks as follows

$$f(\mathbf{W}_i) = \frac{1}{S}\sum_{k=1}^{S}\left[\sum_{l=1}^{O}\{t_{\rm kl} - p_{\rm kl}(\mathbf{W}_i)\}^2\right] \tag{7}$$

where f is the fitness value, $t_{\rm kl}$ is the target output; $p_{\rm kl}$ is the predicted output based on $\mathbf{W}_{\rm i}$; S is the number of training set samples; and, O is the number of output neurons.

6. Application case

In this application case, the model is used to study and predict the outcome of construction claims in Hong Kong. The data from 1991 to 2000 are organized case to case and the dispute characteristics and court decisions are correlated. In total, 1105 sets of construction-related cases were available, of

Table 1
Examples of the input neurons for cases with different types of contract

Input neuron	Cases		
	Remeasurement	Lump sum	Design and build
Type of contract — remeasurement	1	0	0
Type of contract — lump sum	0	1	0
Type of contract — design and build	0	0	1

which 550 from years 1991 to 1995 were used for training, 275 from years 1996 to 1997 were used for testing, and 280 from years 1998 to 2000 were used to validate the network results with the observations.

Through a sensitivity analysis, 13 case elements that seem relevant in courts' decisions are identified. They are, namely, type of contract, contract value, parties involved, type of plaintiff, type of defendant, resolution technique involved, late payment, radical changes in scope, directed changes, constructive changes, liquidated damages involved, legal interpretation of contract documents, and misrepresentation of site. Binary format is adopted for those case elements that can be expressed conveniently in this format; for example, the input element 'liquidated damages involved' receives a 1 if the claim involves liquidated damages or a 0 if it does not.

For those elements that are defined by several alternatives, for example, 'type of contract' could be remeasurement contract, lump sum contract, or design and build contract, they are split into separate input elements, one for each alternative. Each alternative is represented in a binary format, such as 1 for remeasurement contract and 0 for the others if the type of contract is not remeasurement. In that case, only one of these input elements will have a 1 value and all the others will have a 0 value. In this way, the 13 elements are converted into an input layer of 30 neurons, all expressed in binary format. Table 1 shows examples of the input neurons for cases with different types of contract. In the output layer, the court decisions are organized into 6 neurons, namely, client, contractor, engineer, sub-contractor, supplier, and other third parties, and are expressed in binary format also.

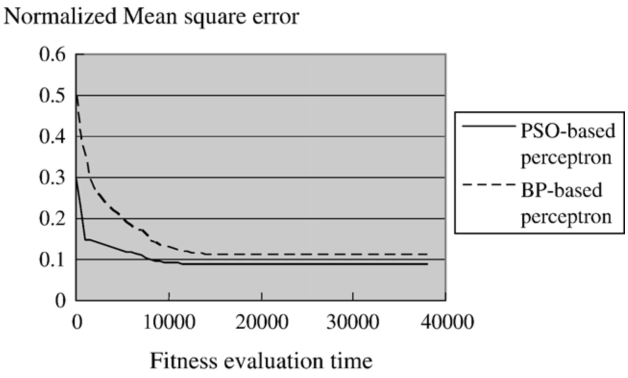

Fig. 1. Relationships between the normalized mean square error and fitness evaluation time during training for PSO-based and BP-based perceptrons.

In order to determine the best parametric architecture for this case, sensitivity analysis is performed. Finally, a perceptron with an input layer with thirty neurons, a hidden layer with fifteen neurons, and output layer with six neurons, is adopted. Moreover, the maximum and minimum velocity values are 0.25 and −0.25, respectively whilst the number of population is 40. The back-propagation with Levenberg–Marquardt (LM) algorithm under the neural network toolbox in MATLAB software [21] is employed as the benchmarking tool for comparison. LM optimization technique is a commonly used ANN that has attained certain improvements such as convergence rates over the original BP algorithm. Details of the ANN algorithm can be found in Hagan and Menhaj [22].

7. Results and discussions

The performance of the PSO-based multi-layer ANN is benchmarked with a conventional BP-based network. Fig. 1 shows the relationships between the normalized mean square error and fitness evaluation time during training for PSO-based and BP-based perceptrons. The fitness evaluation time here for the PSO-based perceptron is equal to the product of the population with the number of generations. Table 2 shows comparisons of the results of network for the two different perceptrons. In the comparison, in order to furnish a comparable initial state, the training process of the BP-based perceptron commences from the best initial population of the corresponding PSO-based perceptron.

It is noted that testing cases of the PSO-based network are able to give a successful prediction rate of up to 80%, which is much higher than by pure chance. Moreover, the PSO-based perceptron exhibits much better and faster convergence performance in the training process as well as better prediction ability in the validation process than those by the BP-based perceptron. From Table 2 and Fig. 1, it can be observed that drawbacks in terms of accuracy (the ability to locate global minimum instead of local minimum) and convergence speed are improved by about 12% and 33% respectively, in comparison to the most recent and modified BP algorithm. It can be concluded that the PSO-based perceptron performs better than the BP-based perceptron. This will furnish the involving parties an alternative in assessing whether or not to take the case to litigation with a much higher confidence. With the assistance of this tool, the number of disputes is expected to be decreased significantly because the cases with lower chances of success will be abandoned more readily. Moreover, attorneys would tailor their cases to the network in order to maximize the chances of entering court and of winning their case. It is

Table 2
Comparison of prediction results for outcome of construction litigation

Algorithm	Training		Validation	
	Coefficient of correlation	Prediction rate	Coefficient of correlation	Prediction rate
BP-based	0.956	0.69	0.953	0.67
PSO-based	0.987	0.81	0.984	0.80

probable that the ANN would need to be adjusted in response to the lawyer's behavior, resulting in modification of their strategies, ad infinitum.

In this study, the binary coding, similar to Arditi et al. [1], is adopted. However, Cohen and Cohen's dummy-coding technique might be more efficient since it can encode an n-dimensional array in $(n-1)$ variables, which might be studied in future works.

Moreover, PSO-based ANN can be applied to different optimization problem. It is demonstrated through this case study that it can serve the purpose of improving the performance of a neural network in the analysis of construction claim outcomes. Other examples having similar characteristics with construction claim outcomes might also be suitable for demonstration of the power of the PSO-based ANN.

8. Conclusions

This paper presents the application of a novel PSO-based perceptron approach for prediction of outcomes of construction litigation according to the characteristics of the individual dispute and the corresponding past court decisions. The optimization algorithm is demonstrated to be able to provide model-free estimates in deducing the output from the input. It is demonstrated from the training and verification simulation that the prediction results of outcomes of construction litigation are more accurate and are obtained in relatively short computational time, when compared with the conventional BP-based perceptron. The final network presented in this study is recommended as an approximate prediction tool for the parties in dispute, since the rate of prediction is up to 80%, which is much higher than chance. It is, of course, recognized that there are limitations in the assumptions used in this study. Other factors that may have certain bearing such as cultural, psychological, social, environmental, and political factors have not been considered here.

References

[1] D. Arditi, F.E. Oksay, O.B. Tokdemir, Predicting the outcome of construction litigation using neural networks, Computer Aided Civil and Infrastructure Engineering 13 (2) (1998) 75–81.

[2] K. Thirumalaiah, M.C. Deo, River stage forecasting using artificial neural networks, Journal of Hydrologic Engineering. ASCE 3 (1) (1998) 26–32.

[3] K.W. Chau, C.T. Cheng, Real-time prediction of water stage with artificial neural network approach, Lecture Notes in Artificial Intelligence 2557 (2002) 715–715.

[4] C.T. Cheng, K.W. Chau, Y.G. Sun, J.Y. Lin, Long-term prediction of discharges in Manwan Reservoir using artificial neural network models, Lecture Notes in Computer Science 3498 (2005) 1040–1045.

[5] J.Y. Lin, C.T. Cheng, K.W. Chau, Using support vector machines for long-term discharge prediction, Hydrological Sciences Journal 51 (4) (2006) 599–612.

[6] N. Muttil, K.W. Chau, Neural network and genetic programming for modelling coastal algal blooms, International Journal of Environment and Pollution 28 (3–4) (2006) 223–238.

[7] C.L. Wu, K.W. Chau, A flood forecasting neural network model with genetic algorithm, International Journal of Environment and Pollution 28 (3–4) (2006) 261–273.

[8] J.X. Xie, C.T. Cheng, K.W. Chau, Y.Z. Pei, A hybrid adaptive time-delay neural network model for multi-step-ahead prediction of sunspot activity, International Journal of Environment and Pollution 28 (3–4) (2006) 364–381.

[9] K.W. Chau, Reliability and performance-based design by artificial neural network, Advances in Engineering Software 38 (3) (2007) 145–149.

[10] R. Govindaraju, A. Rao, Artificial Neural Networks in Hydrology, Kluwer Academic Publishers, Dordrecht, 2000.

[11] S.Y. Liong, W.H. Lim, G.N. Paudyal, River stage forecasting in Bangladesh: neural network approach, Journal of Computing in Civil Engineering, ASCE 14 (1) (2000) 1–8.

[12] J. Kennedy, R. Eberhart, Y. Shi, Swarm Intelligence, Morgan Kaufmann Publishers, San Francisco, 2001.

[13] K.W. Chau, Resolving construction disputes by mediation: Hong Kong experience, Journal of Management in Engineering, ASCE 8 (4) (1992) 384–393.

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

[15] J. Kennedy, The particle swarm: social adaptation of knowledge, Proceedings of the 1997 International Conference on Evolutionary Computation. Indianapolis, 1997, pp. 303–308.

[16] M. Clerc, J. Kennedy, The particle swarm-explosion, stability, and convergence in a multidimensional complex space, IEEE Transactions on Evolutionary Computation 6 (1) (2002) 58–73.

[17] K.W. Chau, River stage forecasting with particle swarm optimization, Lecture Notes in Computer Science 3029 (2004a) 1166–1173.

[18] K.W. Chau, Rainfall-runoff correlation with particle swarm optimization algorithm, Lecture Notes in Computer Science 3174 (2004b) 970–975.

[19] K.W. Chau, Algal bloom prediction with particle swarm optimization algorithm, Lecture Notes in Artificial Intelligence 3801 (2005) 645–650.

[20] K.W. Chau, Particle swarm optimization training algorithm for ANNs in stage prediction of Shing Mun River, Journal of Hydrology 329 (3–4) (2006) 363–367.

[21] M.T. Hagan, M.B. Menhaj, Training feedforward networks with the Marquardt algorithm, IEEE Transactions on Neural Networks 5 (6) (1994) 989–993.

[22] MATLAB, Ver. 6.1.0.450 Release 12.1, Neural network toolbox for use with Matlab. User's Guide, 2001.