Expert Systems with Applications 30 (2006) 519–526

Expert Systems with Applications

www.elsevier.com/locate/eswa

Artificial neural networks with evolutionary instance selection for financial forecasting

Kyoung-jae Kim *

Department of Information Systems, Dongguk University, 3-26, Pil-dong, Chung-gu, Seoul 100-715, South Korea

Abstract

In this paper, I propose a genetic algorithm (GA) approach to instance selection in artificial neural networks (ANNs) for financial data mining. ANN has preeminent learning ability, but often exhibit inconsistent and unpredictable performance for noisy data. In addition, it may not be possible to train ANN or the training task cannot be effectively carried out without data reduction when the amount of data is so large. In this paper, the GA optimizes simultaneously the connection weights between layers and a selection task for relevant instances. The globally evolved weights mitigate the well-known limitations of gradient descent algorithm. In addition, genetically selected instances shorten the learning time and enhance prediction performance. This study applies the proposed model to stock market analysis. Experimental results show that the GA approach is a promising method for instance selection in ANN.

Keywords: Instance selection; Genetic algorithms; Artificial neural networks; Financial forecasting

1. Introduction

In general, artificial neural networks (ANNs) can produce robust performance when a large amount of data is available. However, ANN often exhibits inconsistent and unpredictable performance on noisy data. In addition, it may not be possible to train ANN or the training task cannot be effectively carried out without data reduction when a data set is too huge. Data reduction can be achieved in many ways such as feature selection or feature discretization (Blum & Langley, 1997; Kim & Han, 2000; Liu & Motoda, 1998).

One facet of data mining concerns the selection of relevant instances for this reason. Instances are a collection of training examples in supervised learning and instance selection chooses a part of the data that is representative and relevant to the characteristics of all the data. Instance selection is one of popular methods for dimensionality reduction and is directly related to data reduction. Although instance selection is the most complex form of data reduction because the computationally expensive prediction methods must be invoked more often to determine the effectiveness of instance selection, we can usually remove irrelevant instances as well as noise and redundant data (Liu & Motoda, 2001; Weiss & Indurkhya, 1998).

Many researchers have suggested instance selection methods such as squashed data, critical points, prototype construction, in addition to many forms of sampling (Liu & Motoda, 2001). The efforts to select relevant instances from an initial data set have stemmed from the need to reduce immense storage requirements and computational loads (Kuncheva, 1995). The other perspective on this subject, as pointed out in Dasarathy (1990), is to achieve enhanced performance from the learning algorithm through instance selection. In addition, training time may be shortened by use of the proper instance selection algorithm.

This paper proposes a new hybrid model of ANN and genetic algorithms (GAs) for instance selection. An evolutionary instance selection algorithm reduces the dimensionality of data and may eliminate noisy and irrelevant instances. In addition, this study simultaneously searches the connection weights between layers in ANN through an evolutionary search. The genetically evolved connection weights mitigate the well-known limitations of gradient descent algorithm.

The rest of this paper is organized as follows: Section 2 presents the research background. Section 3 proposes the evolutionary instance selection algorithm and describes the benefits of the proposed algorithm. Section 4 describes the application of the proposed algorithm. Conclusions and the limitations of this study are presented in Section 5.

* Tel.: +82 2 2260 3324; fax: +82 2 2260 3684.
E-mail address: kjkim@dongguk.edu

0957-4174/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2005.10.007

2. Research background

For some applications, quality of data mining is improved with additional instances. However, the number of instances may tend to increase the complexity of induced solution. Increased complexity is not desirable, but may be the price to pay for better performance. In addition, increased complexity decreases the interpretability of the result (Weiss & Indurkhya, 1998). In this sense, many researchers have suggested instance selection methods. The following sections present some instance selection methods as described by prior research.

2.1. Instance selection methods

Instance-based learning algorithms often faced the problem of deciding which instances to store for use during generalization in order to avoid excessive storage and time complexity, and to improve generalizability by avoiding noise and overfitting (Wilson & Martinez, 2000). Many researchers have addressed the problem of training data reduction and have presented algorithms for maintaining an instance base or case base in instance-based learning algorithms.

Kuncheva (1993) classified instance selection techniques (or editing techniques) into the following three categories: Condensed Nearest Neighbor rule, Generated or Modified Prototypes, and Two-Level Classifiers. The following presents some basic concepts of each category as described by prior research. A detailed explanation may be found in the references of this paper.

2.1.1. Condensed nearest neighbor rule

Hart (1968) made one of the first attempts to develop an instance selection rule. Hart's algorithm, the Condensed Nearest Neighbor rule, finds a subset S of the training set T such that every member of T is closer to a member of S of the same class than to a member of S of a different class. Subsequent work extended Hart's algorithm, specifically the Selective Nearest Neighbor rule (Ritter, Woodruff, Lowry, & Isenhour, 1975) and the Reduced Nearest Neighbor rule (Gates, 1972). In addition, Wilson (1972) introduced the Edited Nearest Neighbor algorithm and Tomek (1976) proposed the All k-NN method of editing.

2.1.2. Generated or modified prototypes

This category is composed of techniques that establish new prototypes or adjust a limited number of instances. A large group of studies within this category are implemented by ANN including feature-map classifiers, learning vector quantiziers (Kuncheva, 1995).

2.1.3. Two-level classifiers

This category employs two or more classifiers and allocates a part of all instances to the classifier, which appears most appropriate. Tetko and Villa (1997) proposed the Efficient Partition algorithm, which is used to obtain an efficient partition of noisy instances, whose distribution is proportional to the complexity of the analyzed function. This is to focus the training of ANN on the most complex and informative domains of the data set and accelerate the learning phase. They concluded that the efficiently partitioned instances enhance the predictability of ANN in comparison with a random selection of instances. Oh and Han (2000) proposed the integrated neural network model using change-point detection. They partitioned instances according to each detected change-point, and then applied each partitioned instance to each ANN of multiple ANN.

Instance selection in instance-based learning algorithms may be considered as a method of knowledge refinement and it maintains the instance-base. In this sense, some researchers proposed many instance selection algorithms for maintaining the case-base in case-based reasoning (CBR) systems. Smyth (1998) presented an approach to maintenance, which is based on the deletion of harmful and redundant cases from the case-base. In addition, McSherry (2000) suggested an instance selection method in the construction of a case library in which evaluation of the coverage contributions of candidate instances are based on an algorithm called *disCover*. This algorithm reverses the direction of CBR to discover all cases that can be solved with a given case-base.

Although many different approaches have been used to address the problem of case authoring and data explosion for instance-based algorithms, there is little research on instance selection in ANN. Reeves and Taylor (1998) suggested that a GA is a promising approach to finding 'better' training data set for classification problems in radial basis function (RBF) nets. Reeves and Bush (2001) reported that the GA can also be used effectively to find a smaller subset of a 'good' training set in RBF nets for both classification and regression problems. Although, the GA has been shown to be a promising instance selection method for RBF nets, its performances on other neural network models are untested.

2.2. Genetic algorithms

The GA has been investigated recently and shown to be effective in exploring a complex space in an adaptive way, guided by the biological evolution mechanisms of *selection*, *crossover*, and *mutation* (Adeli & Hung, 1995). The GA simulates the mechanics of population genetics by maintaining a population of knowledge structure, which is made to evolve (Odetayo, 1995).

The problems must be represented in a suitable form to be handled by the GA. The GA often works with a form of binary coding. If the problems are coded as chromosomes, the population is initialized. Each chromosome within the population is gradually evolved by biological operations. Once the population size is chosen, the initial population is randomly generated (Bauer, 1994). After the initialization step, each chromosome is evaluated by the fitness function. According to the value of the fitness function, the chromosomes associated with the fittest individuals will be reproduced more often than those associated unfit individuals (Davis, 1994).

The GA works with three operators that are iteratively used. The *selection* operator determines which individuals may survive (Hertz & Kobler, 2000). The *crossover* operator allows the search to fan out in diverse directions looking for attractive solutions and permits chromosomal material from different parents to be combined in a single child. In addition, the mutation operator arbitrarily alters one or more components of a selected chromosome. It provides the means for introducing new information into the population. Finally, the GA tends to converge on optimal or near-optimal solutions (Wong & Tan, 1994).

The GA is usually employed to improve the performance of artificial intelligence techniques. For ANN, the GA was applied to the selection of neural network topology including optimizing a relevant feature subset, determining the optimal number of hidden layers and processing elements. In addition, some researchers searched the connection weights of ANN using the GA instead of local search algorithms including a gradient descent algorithm. They suggested that global search techniques including the GA might prevent ANN from falling into a local optimum (Gupta & Sexton, 1999; Kim & Han, 2000; Sexton, Dorsey, & Johnson, 1998).

2.3. Prior research on stock market prediction using ANN

Many studies on stock market prediction using artificial intelligence (AI) techniques have been performed during the past decade. The early days of these studies focused on estimating the level of the return on stock price index. One of the earliest studies, Kimoto, Asakawa, Yoda, and Takeoka (1990) used several learning algorithms and prediction methods for developing a prediction system for the Tokyo Stock Exchange Prices Index. They used the modular neural network to learn the relationships among various market factors. They concluded that the correlation coefficient produced by their model is much higher than that produced by multiple regression. However, the correlation coefficient may not be a proper measure for prediction performance. Kamijo and Tanikawa (1990) used the recurrent neural network for analyzing candlestick charts. A candlestick chart is a Japanese style chart used to visualize stock price patterns. In these studies, they did not perform any statistical test for the significance of the empirical results.

Some researchers investigated the issue of predicting the stock index futures market. Choi, Lee and Lee (1995) and Trippi and DeSieno (1992) predicted the daily direction of change in the S&P 500 index futures using ANN. Trippi and DeSieno (1992) combined the outputs of individual networks using logical (Boolean) operators to produce a set of composite rules. They suggested that their best composite synthesized rule set system achieved a higher gain than previous research. Choi et al. (1995) compared their approach with previous study and suggested that they earned a higher annualized gain than the previous study. However, the annualized gain may not be an appropriate measure for prediction performance because it varies according to the fee for trade and the trading strategy. Duke and Long (1993) predicted German government daily bond futures using backpropagation (BP) neural networks. They reported that the 53.94% of the patterns are accurately predicted through the moving simulation method. Most of the above studies simply applied ANN to stock market prediction.

Recent research tends to hybridize several AI techniques. Nikolopoulos and Fellrath (1994) developed a hybrid expert system for investment advising. In their study, genetic algorithms were used to train and configure the architecture of investor's neural network component. Hiemstra (1995) proposed fuzzy expert systems to predict stock market returns. He suggested that ANN and fuzzy logic could capture the complexities of functional mapping because they do not require the specification of the function to approximate. Some researchers tend to include novel factors for the learning process. Kohara, Ishikawa, Fukuhara and Nakamura (1997) incorporated prior knowledge to improve the performance of stock market prediction. Prior knowledge in their study included non-numerical factors such as political and international events. They made use of prior knowledge of stock price predictions and newspaper information on domestic and foreign events. A more recent study of Lee and Jo (1999) developed an expert system, which uses knowledge in a candlestick chart analysis. The expert system had patterns and rules, which could predict future stock price movements. The experimental results revealed that a developed knowledge-base could provide excellent indicators. In addition, Tsaih, Hsu and Lai (1998) integrated a rule-based technique and ANN to predict the direction of change of the S&P 500 stock index futures on a daily basis.

Stock market data, however, includes tremendous noise and non-stationary characteristics; thus, the training process for ANN tends to be difficult. In addition, the possibility of local convergence of the gradient search techniques may be another difficulty for learning patterns.

3. A GA approach to instance selection for ANN

As mentioned earlier, there are many studies on instance selection for the instance-based learning algorithm. However, there are few studies on instance selection for ANN. Thus, there are few relevant theories concerning instance selection for ANN. This paper proposes the GA approach to instance selection for ANN (GAIS). The overall framework of GAIS is shown in Fig. 1. In this study, the GA supports the simultaneous optimization of connection weights and selection of relevant instances.

The algorithm of GAIS consists of the following three phases: GA search phase, feed-forward computation phase, and validation phase.

3.1. GA search phase

In the GA search phase, the GA searches the search space to find optimal or near-optimal connection weights and relevant instances for ANN. The populations, the connection weights and the codes for instance selection, are initialized into random values before the search process. The parameters for searching

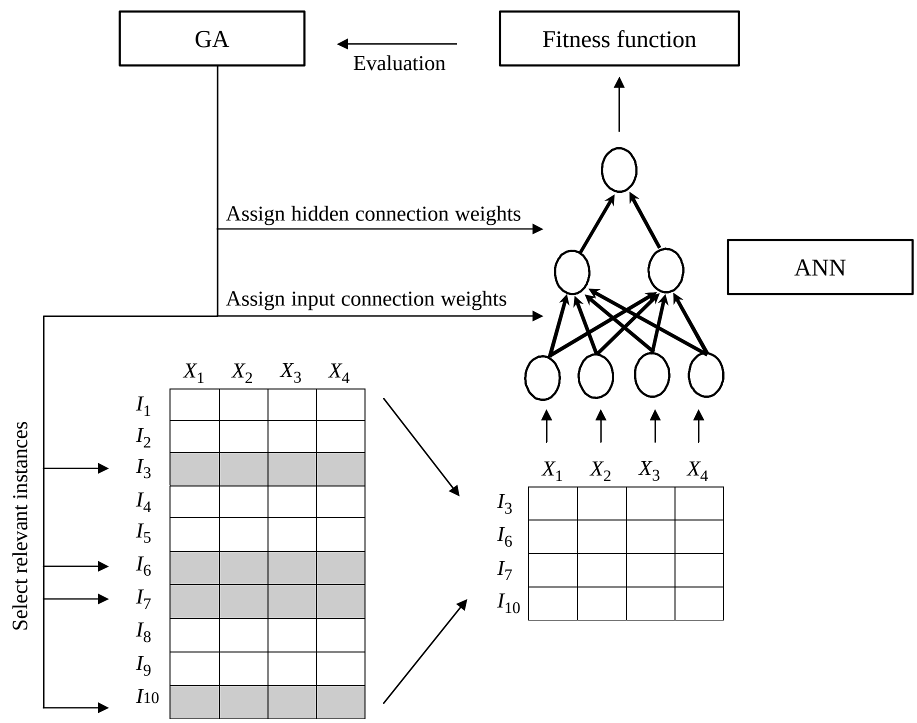

Fig. 1. Overall framework of GAIS.

must be encoded on chromosomes. This study needs three sets of parameters. The first set is the set of connection weights between the input layer and the hidden layer of the network. The second set is the set of connection weights between the hidden layer and the output layer. As mentioned earlier, the above two sets may mitigate the limitation of the gradient descent algorithm. The third set represents the codes for instance selection.

The strings have the following encoding: each processing element in the hidden layer receives signals from the input layer. The first set of bits represents the connection weights between the input layer and the hidden layer. Each processing element in the output layer receives signals from the hidden layer. The next set of bits indicates the connection weights between the hidden layer and the output layer. The following bits are instance selection codes for the training data. The parameters to be searched use only the information about the selected instances within the training data. In this phase, the GA operates the process of crossover and mutation on initial chromosomes and iterates until the stopping conditions are satisfied.

3.2. Feed-forward computation phase

This phase is the process of feed-forward computation in ANN. Proper activation function is required to facilitate the learning process. However, there are no clear criteria regarding which activation function to use. Some researchers recommended the sigmoid function for classification problems and the hyperbolic tangent function for forecasting problems because of the difference between the sigmoid and the hyperbolic tangent function for the value range of delta weights with the SSE error function (Coakley & Brown, 2000). In addition, the majority of back-propagation applications used the sigmoid activation function (Hansen, McDonald, & Nelson, 1999). There are few comparative studies between the sigmoid function and other activation functions in ANN. This study uses the sigmoid function as the activation function because this study is performed to classify the accurate direction of change in the daily stock price index. The linear function is used as a combination function for the feed-forward computation with the derived connection weights from the first phase.

3.3. Validation phase

The derived connection weights are applied to the holdout data. This phase is indispensable to validate the generalizability because ANN has the eminent ability of learning the known data. Table 1 summarizes the algorithms of GAIS.

4. Application: analysis of the stock market data

This section applies GAIS to stock market prediction. The efficiency and effectiveness of GAIS may be properly tested because the stock market data is very noisy and complex. Many studies on stock market prediction using artificial intelligence techniques were performed in the past decade. Some of them, however, did not produce outstanding prediction accuracy partly because of the tremendous noise and non-stationary

Table 1
The algorithms of GAIS

Step 0	Initialize the populations (the connection weights between layers and the codes for instance selection). (Set to small random values between 0.0 and 1.0)
Step 1	If stopping condition is false, do Step 2. Otherwise, stop the process
Step 2	Do Steps 3–9
Step 3	Each processing element in the input layer receives an input signal and forwards this signal to all processing elements in the hidden layer
Step 4	Each processing element in the hidden layer sums its weighted input signals and applies the sigmoid activation function to compute its output signal of the hidden processing element and forwards it to all processing elements in the output layer
Step 5	Each processing element in the output layer sums its weighted signals from the hidden layer and applies the sigmoid activation function to compute its output signal of the output processing element and computes the difference between the output signal and the target value
Step 6	Calculate fitness. (Fitness function: Average predictive accuracy on the selected instances within the training data)
Step 7	Select individuals to become parents of the next generation
Step 8	Create a second generation from the parent pool. (Perform crossover and mutation)
Step 9	Test stopping condition and go back to Step 1

characteristics in stock market data. If these factors are not appropriately controlled, the prediction system does not produce significant performance. When the prediction is executed using long-term data, this is more important to manage the consistency of prediction.

4.1. Application data

The application data used in this study consists of technical indicators and the direction of change in the daily Korea stock price index (KOSPI). The total number of samples is 2348 trading days, from January 1991 to December 1998. This study divides the samples into eight data sets according to the trading year. Experiments are repeated eight times for each data set to reflect specific knowledge as time passes.

The direction of daily change in the stock price index is categorized as ‘0’ or ‘1’. ‘0’ means that the next day’s index is lower than the today’s index, and ‘1’ means that the next day’s index is higher than today’s index. I select 12 technical indicators as feature subsets by the review of domain experts and prior research. Table 2 gives selected features and their formulas.

4.2. Experiments

The following experiments are carried out:

4.2.1. Whole training data

The whole training samples are used as the training data. This is the conventional method of data analysis.

4.2.2. Selected instances with GAIS

Experiments on stock market data are implemented using GAIS. The procedure of the experiment is as follows. The GA searches for optimal or near-optimal connection weights and relevant instances for ANN. As mentioned earlier, this study needs three sets of parameters: The connection weights between the input and the hidden layer, the connection weights between the hidden and the output layer, and the codes for instance selection.

This study uses the following encoding for the strings: 12 input features are used and 12 processing elements in the hidden layer are employed. Each processing element in the hidden layer receives 12 signals from the input layer. The first 144 bits represent the connection weights between the input

Table 2
Selected features and their formulas

Names of feature	Formula
Stochastic %K	$(C_t-\mathrm{LL}_{t-5})/(\mathrm{HH}_{t-5}-\mathrm{LL}_{t-5})\times 100$
Stochastic %D	$\sum_{i=0}^{n-1}\%K_{t-i}/n$
Stochastic slow %D	$\sum_{i=0}^{n-1}\%D_{t-i}/n$
Momentum	C_t-C_{t-4}
ROC (rate of change)	$(C_t/C_{t-n})\times 100$
LW %R (Larry William’s %R)	$(H_n-C_t)/(H_n-L_n)\times 100$
A/D Oscillator (accumulation/distribution oscillator)	$(H_t-C_{t-1})/(H_t-L_t)$
Disparity 5 days	$(C_t/\mathrm{MA}_5)\times 100$
Disparity 10 days	$(C_t/\mathrm{MA}_{10})\times 100$
OSCP (price oscillator)	$(\mathrm{MA}_5-\mathrm{MA}_{10})/\mathrm{MA}_5$
CCI (commodity channel index)	$(M_t-\mathrm{SM}_t)/(0.015\times D_t)$
RSI (relative strength index)	$100-100/\left(1+\left(\sum_{i=0}^{n-1}\mathrm{Up}_{t-i}/n\right)/\left(\sum_{i=0}^{n-1}\mathrm{Dw}_{t-i}/n\right)\right)$

C, closing price; L, low price; H, high price; LL_n, lowest low price in the last n days; HH_n, highest high price in the last n days; M, moving average of price; M_t, $(H_t+L_t+C_t)/3$; SM_t, $\sum_{i=1}^{n}M_{t-i+1}/n$; D_t, $\sum_{i=1}^{n}|M_{t-i+1}-\mathrm{SM}_t|/n$; Up, upward price change; Dw, downward price change.

Table 3
Number of instances

Set	Year								Total
	1991	1992	1993	1994	1995	1996	1997	1998	
Training instances for GANN	234	236	237	237	235	235	234	234	1882
Selected instances for GAIS	74	71	87	66	93	86	93	85	655
Holdout instances for GANN & GAIS	58	58	59	59	58	58	58	58	466

layer and the hidden layer. These bits are searched from −5 to 5. Each processing element in the output layer receives signals from the hidden layer. The next 12 bits indicate the connection weights between the hidden layer and the output layer. These bits also varied between −5 and 5. The following bits are instance selection codes for the training data. The chromosome of these bits consists of n genes (where n is the number of initial training instances), each one with two possible states: 0 or 1. '1' means the associated instance is selected into the analysis and '0' means the associated instance is not chosen.

The encoded chromosomes are searched to maximize the fitness function. The fitness function is specific to applications. In this study, the objectives of the model are to approximate connection weights and to select relevant instances for the correct solutions. These objectives can be represented by the average prediction accuracy of the selected instances within the training data. Thus, this study applies the average prediction accuracy of the selected instances within the training data to the fitness function. Mathematically, the fitness function is represented as Eq. (1):

$$\text{Fitness} = \frac{1}{n}\sum_{i=1}^{n} \mathrm{CR}_i \quad (i = 1, 2, \ldots, n)$$
$$\begin{cases} \text{if } \mathrm{PO}_i = \mathrm{AO}_i & \mathrm{CR}_i = 1 \\ \text{otherwise} & \mathrm{CR}_i = 0 \end{cases} \tag{1}$$

where CR_i is the prediction result for the ith trading day which is denoted by 0 or 1, PO_i is the predicted output from the model for the ith trading day, and AO_i is the actual output for the ith trading day.

For the controlling parameters of the GA search, the population size is set at 100 organisms and the crossover and mutation rates are varied to prevent ANN from falling into a local minimum. The value of the crossover rate is set at 0.7 while the mutation rate is 0.1. For the crossover method, the uniform crossover method is considered better at preserving the schema, and can generate any schema from the two parents, while single-point and two-point crossover methods may bias the search with the irrelevant position of the variables. Thus, this study performs crossover using the uniform crossover routine. For the mutation method, this study generates a random number between 0 and 1 for each of the variables in the organism. If a variable gets a number that is less than or equal to the mutation rate, then that variable is mutated. As the stopping condition, only 100 generations are permitted.

4.3. Experimental results and discussions

This study compares GAIS to the conventional ANN with the GA. The conventional ANN with the GA, named GANN, denotes the ANN model with the connection weights, which are determined by the GA. This model does not use the gradient descent algorithm but uses the GA to determine the connection weights between layers. However, this model analyzes all available training data to learn. On the other hand, GAIS also uses the GA to determine the connection weights, but learns the patterns of the stock market data from the selected instances through an evolutionary search. For the GANN model, about 20% of the data is used for holdout and 80% for training. The training data is used to search for the optimal or near-optimal parameters and is employed to evaluate the fitness function. The holdout data is used to test the results with the data that is not utilized to develop the model. The number of the training instances in GANN and the number of the selected instances within the training instances in GAIS for each year are presented in Table 3.

Table 4 describes the average prediction accuracy of each model.

In Table 4, GAIS outperforms GANN by 6.22% for the holdout data. In addition, GAIS has higher accuracy than GANN by 10.74% for the training data. This result may be caused by the benefits of the instance selection through evolutionary search techniques.

The McNemar tests are used to examine whether GAIS significantly outperforms GANN. This test may be used with nominal data and is particularly useful with a before–after measurement of the same subjects (Cooper & Emory, 1995). The McNemar value and its p values of the holdout data are

Table 4
Average predictive performance (hit ratio: %)

Year	GANN		GAIS	
	Training	Holdout	Training	Holdout
1991	63.68	53.45	74.32	72.41
1992	64.83	56.90	77.46	58.62
1993	61.18	59.32	70.11	59.32
1994	62.87	57.63	74.24	61.02
1995	69.36	65.52	81.72	67.24
1996	65.11	65.52	76.74	77.59
1997	64.96	58.62	65.59	58.62
1998	61.11	56.90	78.82	68.97
Total	64.13%	59.23%	74.87%	65.45%

5.262 and 0.022, respectively. This means that GAIS performs better than GANN at the 5% statistical significance level.

5. Concluding remarks

Prior studies tried to optimize the controlling parameters of ANN using global search algorithms. Some of them only focused on the optimization of the connection weights of ANN. Others placed little emphasis on the optimization of the learning algorithm itself, but most studies focused little on instance selection for ANN. In this paper, I use the GA for ANN in two ways. I first use the GA to determine the connection weights between layers. This may mitigate the well-known limitations of the gradient descent algorithm. In addition, I adopt the evolutionary instance selection algorithm for ANN. This directly removes irrelevant and redundant instances from the training data. I conclude that GA-based learning and the instance selection algorithm (GAIS) significantly outperforms the conventional GA-based learning algorithm (GANN).

The prediction performance may be more enhanced if the GA is employed not only for instance selection but also for relevant feature selection, and this remains a very interesting topic for further study. Although instance selection is a direct method of noise and dimensionality reduction, feature selection effectively reduces the dimensions of feature space. In addition, while ANN performed well with GA-based learning and instance selection, other instance-based learning algorithms including CBR may also prove effective in place of ANN. Of course, there are still many tasks to be done for GAIS. The generalizability of GAIS should be further tested by applying it to other problem domains.

References

Adeli, H., & Hung, S. (1995). *Machine learning: Neural networks, genetic algorithms, and fuzzy systems*. New York: Wiley.

Bauer, R. J. (1994). *Genetic algorithms and investment strategies*. New York: Wiley.

Blum, A., & Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence*, *97*(1–2), 245–271.

Choi, J.H., Lee, M.K., & Lee, M.W. (1995). Trading S&P 500 stock index futures using a neural network. *Proceedings of the third annual international conference on artificial intelligence applications on wall street* (pp. 63–72). New York.

Coakley, J. R., & Brown, C. E. (2000). Artificial neural networks in accounting and finance: Modeling issues. *International Journal of Intelligent Systems in Accounting, Finance and Management*, *9*, 119–144.

Cooper, D. R., & Emory, C. W. (1995). *Business research methods*. Chicago: Irwin.

Dasarathy, B. V. (1990). *Nearest neighbor (NN) norms: NN pattern classification techniques*. California: IEEE Computer Society Press.

Davis, L. (1994). Genetic algorithms and financial applications. In G. J. Deboeck (Ed.), *Trading on the edge* (pp. 133–147). New York: Wiley.

Duke, L. S., & Long, J. A. (1993). Neural network futures trading—a feasibility study. In: Society for worldwide interbank financial telecomunications, adaptive intelligent systems (pp. 121–132). Amsterdam: Elsevier.

Gates, G. W. (1972). The reduced nearest neighbor rule. *IEEE Transactions on Information Theory*, *18*(3), 431–433.

Gupta, J. N. D., & Sexton, R. S. (1999). Comparing backpropagation with a genetic algorithm for neural network training. *Omega*, *27*(6), 679–684.

Hansen, J. V., McDonald, J. B., & Nelson, R. D. (1999). Time series prediction with genetic-algorithm designed neural networks: An empirical comparison with modern statistical models. *Computational Intelligence*, *15*(3), 171–184.

Hart, P. E. (1968). The condensed nearest neighbor rule. *IEEE Transactions on Information Theory*, *14*, 515–516.

Hertz, A., & Kobler, D. (2000). A framework for the description of evolutionary algorithms. *European Journal of Operational Research*, *126*(1), 1–12.

Hiemstra, Y. (1995). Modeling structured nonlinear knowledge to predict stock market returns. In R. R. Trippi (Ed.), *Chaos & nonlinear dynamics in the financial markets: Theory, evidence and applications* (pp. 163–175). Chicago, IL: Irwin.

Kamijo, K., & Tanigawa, T. (1990). Stock price pattern recognition: A recurrent neural network approach. *Proceedings of the international joint conference on neural networks* (pp. 215–221). San Diego, CA.

Kim, K., & Han, I. (2000). Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index. *Expert Systems with Applications*, *19*(2), 125–132.

Kimoto, T., Asakawa, K., Yoda, M., & Takeoka, M. (1990). Stock market prediction system with modular neural network. *Proceedings of the International Joint Conference on Neural Networks* (pp. 1–6). San Diego, CA.

Kohara, K., Ishikawa, T., Fukuhara, Y., & Nakamura, Y. (1997). Stock price prediction using prior knowledge and neural networks. *International Journal of Intelligent Systems in Accounting, Finance and Management*, *6*(1), 11–22.

Kuncheva, L. I. (1993). 'Change-glasses' approach in pattern recognition. *Pattern Recognition Letters*, *14*, 619–623.

Kuncheva, L. I. (1995). Editing for the k-nearest neighbors rule by a genetic algorithm. *Pattern Recognition Letters*, *16*(8), 809–814.

Lee, K. H., & Jo, G. S. (1999). Expert systems for predicting stock market timing using a candlestick chart. *Expert Systems with Applications*, *16*(4), 357–364.

Liu, H., & Motoda, H. (1998). Feature transformation and subset selection. *IEEE Intelligent Systems and Their Applications*, *13*(2), 26–28.

Liu, H., & Motoda, H. (2001). Data reduction via instance selection. In H. Liu, & H. Motoda (Eds.), *Instance selection and construction for data mining* (pp. 3–20). Massachusetts: Kluwer Academic Publishers.

McSherry, D. (2000). Automating case selection in the construction of a case library. *Knowledge Based Systems*, *13*(2–3), 133–140.

Nikolopoulos, C., & Fellrath, P. (1994). A hybrid expert system for investment advising. *Expert Systems*, *11*(4), 245–250.

Odetayo, M. O. (1995). Knowledge acquisition and adaptation: A genetic approach. *Expert Systems*, *12*(1), 3–13.

Oh, K. J., & Han, I. (2000). Using change-point detection to support artificial neural networks for interest rates forecasting. *Expert Systems with Applications*, *19*(2), 105–115.

Reeves, C. R., & Bush, D. R. (2001). Using genetic algorithms for training data selection in RBF networks. In H. Liu, & H. Motoda (Eds.), *Instance selection and construction for data mining* (pp. 339–356). Massachusetts: Kluwer Academic Publishers.

Reeves, C. R., & Taylor, S. J. (1998). Selection of training sets for neural networks by a genetic algorithm. In A. E. Eiden, T. Bäck, M. Schoenauer, & H.-P. Schwefel (Eds.), *Parallel problem-solving from nature-PPSN V*. Berlin: Springer.

Ritter, G. L., Woodruff, H. B., Lowry, S. R., & Isenhour, T. L. (1975). An algorithm for a selective nearest neighbor decision rule. *IEEE Transactions on Information Theory*, *21*(6), 665–669.

Sexton, R. S., Dorsey, R. E., & Johnson, J. D. (1998). Toward global optimization of neural networks: A comparison of the genetic algorithm and backpropagation. *Decision Support Systems*, *22*(2), 171–185.

Smyth, B. (1998). Case-base maintenance. *Proceedings of the 11th international conference on industrial & engineering applications of artificial intelligence & expert systems* (pp. 507–516).

Tetko, I. V., & Villa, A. E. P. (1997). Efficient partition of learning data sets for neural network training. *Neural Networks*, *10*(8), 1361–1374.

Tomek, I. (1976). An experiment with the edited nearest neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, *6*(6), 448–452.

Trippi, R. R., & DeSieno, D. (1992). Trading equity index futures with a neural network. *Journal of Portfolio Management*, *19*, 27–33.

Tsaih, R., Hsu, Y., & Lai, C. C. (1998). Forecasting S&P 500 stock index futures with a hybrid AI system. *Decision Support Systems*, *23*(2), 161–174.

Weiss, S. M., & Indurkhya, N. (1998). *Predictive data mining: A practical guide*. California: Morgan Kaufmann Publishers.

Wilson, D. L. (1972). Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, *2*(3), 408–421.

Wilson, D. R., & Martinez, T. R. (2000). Reduction techniques for instance-based learning algorithms. *Machine Learning*, *38*, 257–286.

Wong, F., & Tan, C. (1994). Hybrid neural, genetic, and fuzzy systems. In G. J. Deboek (Ed.), *Trading on the edge* (pp. 243–261). New York: Wiley.