Journal of Hydrology 367 (2009) 125–137

Contents lists available at ScienceDirect

Journal of Hydrology

journal homepage: www.elsevier.com/locate/jhydrol

Evolutionary artificial neural networks for hydrological systems forecasting

Yung-hsiang Chen [a,b], Fi-John Chang [a,*]

[a] Department of Bioenvironmental Systems Engineering, National Taiwan University, Taiwan, ROC
[b] Water Resources Agency, Ministry of Economic Affairs, Taiwan, ROC

ARTICLE INFO

Article history:
Received 20 June 2008
Received in revised form 8 December 2008
Accepted 12 January 2009

This manuscript was handled by K. Georgakakos Editor-in-Chief with the assistance of Enrique R. Vivoni, Associate Editor.

Keywords:
Evolutionary artificial neural network (EANN)
Genetic algorithm (GA)
Time series
Forecasting
Hydrology
Water resources

SUMMARY

The conventional ways of constructing artificial neural network (ANN) for a problem generally presume a specific architecture and do not automatically discover network modules appropriate for specific training data. Evolutionary algorithms are used to automatically adapt the network architecture and connection weights according to the problem environment without substantial human intervention. To improve on the drawbacks of the conventional optimal process, this study presents a novel evolutionary artificial neural network (EANN) for time series forecasting. The EANN has a hybrid procedure, including the genetic algorithm and the scaled conjugate gradient algorithm, where the feedforward ANN architecture and its connection weights of neurons are simultaneously identified and optimized. We first explored the performance of the proposed EANN for the Mackey–Glass chaotic time series. The performance of the different networks was evaluated. The excellent performance in forecasting of the chaotic series shows that the proposed algorithm concurrently possesses efficiency, effectiveness, and robustness. We further explored the applicability and reliability of the EANN in a real hydrological time series. Again, the results indicate the EANN can effectively and efficiently construct a viable forecast module for the 10-day reservoir inflow, and its accuracy is superior to that of the AR and ARMAX models.

Introduction

In Darwin's concept of survival of the fittest, creatures must be able to adjust to the environment for survival. Learning and evolution are two fundamental forms of adaptation. Learning refers to the process of modifying behaviour to adjust to the environment in different stages for an individual during its life. When learning well an individual is capable of adapting to a similar environment. In contrast with the learning of single individuals, evolution mentions the process of a population of parent species passing genes to their offspring through reproduction, crossover, and mutation for generations. Fig. 1 shows a comparison of an individual's learning and a population's evolution. Among various research fields, artificial neural networks (ANNs) and evolutionary algorithms (EAs) are typical applications of learning and evolution, respectively.

With the ability to process massive information and deal with high non-linearity, ANNs have been widely studied and successfully applied to various fields, e.g., hydrology and water resources, in recent years (Hsu et al., 1995; Sajikumar and Thandaveswara, 1999; Chang et al., 2005, 2007; Karunasinghe and Liong, 2006; Sahoo and Ray, 2006; Chiang et al., 2007; Kim and Kim, 2008). Most present artificial neural networks (ANNs), however, rely heavily on human experts who have sufficient knowledge about the different aspects of the network and the problem domain (Abraham, 2004). These generally presume a specific architecture and do not automatically discover network modules appropriate for specific training data. The selection of network architecture has a significant influence on the performance of ANNs. Too simple a network architecture might not meet the demand for accuracy, while a too complicated architecture might reduce the generalization ability of network due to over-fitting. As the complexity of the problem increases, manual design becomes more difficult. The conventional way to design network architecture involves a destructive algorithm (Abrahart et al., 1998) and a constructive algorithm (Kwok and Yeung, 1997). However, the application by designers of those two algorithms to increase or decrease the hidden layers or neurons is based on a predefined network architecture, which usually misleads the search to a restrictedly structural local optimum (Angeline et al., 1994; Castillo et al., 2007). The major challenge of applying ANN is how to evolve unique neural network architecture and its corresponding weight values for a specific problem.

The genetic algorithm (GA), a branch of EAs, was proposed by Holland (1975) based on Darwin's concept of "survival of the fittest". GA regards optimization problems as natural evolutionary species and transfers the search process into an evolutionary process. Having the abilities of global search and evolutionary adaptation properties, the GA is able to supplement the insufficiency of

* Corresponding author. Tel.: +886 2 23639461; fax: +886 2 23635854.
E-mail address: changfj@ntu.edu.tw (F.-J. Chang).

0022-1694/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.jhydrol.2009.01.009

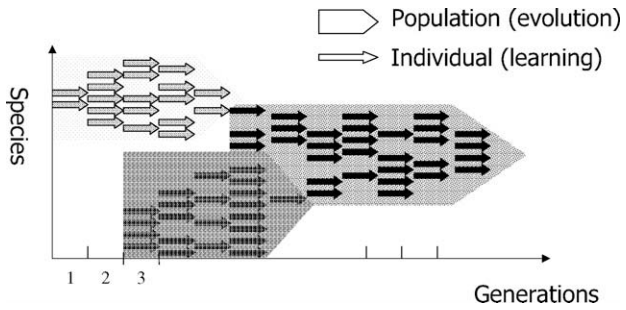


Fig. 1. Comparison of individual's learning and population's evolution.

ANN. In such a way it would couple the evolution of the population and the learning process of each individual, achieving better adaptation of the whole environment to a generic fitness landscape. These investigations led to the birth of a new framework generally referred to as evolutionary artificial neural networks (EANNs). The creative evolutionary systems could have a major impact on the effectiveness and efficiency of designing neural networks. Evolutionary algorithms are used to automatically adapt the connection weights, network architecture and learning rules according to the problem environment without substantial human intervention. The evolution of connection weights is applied to improve the adaptation ability of connection weights by global training based on predefined network architecture. The evolution of network architecture can lead the training to adapt different optimization problems. The objects to be evolved could be only network topology or both network topology and connection weights.

The evolution of ANN's architecture means a process of optimizing parameters of network architecture. The optimization of parameters generally depends on the characteristics of problems, and it is often true that using an ANN to optimize the best network architecture is not easy, especially for highly non-linear data. A better way is to introduce other algorithms of global optimization (e.g., GAs) that would increase the opportunities to search the near-optimal solution. For example, the conventional approach for feedforward ANN is to predefine the network architecture and then implement the iterated process of optimizing connection weights. Different from traditional ANN requiring predefining the network architecture, EANN is able to automatically search for the best network architecture and near-optimal solution as well by evolution and learning more efficiently and effectively. In other words, EANN implements the GA to locate a good region in the space (i.e., evolved network architecture) and then a local search procedure (gradient search) is used to find a near-optimal solution in this region (Yao, 1999).

The main purpose of this study is to propose an EANN for automatically constructing the optimal network architecture and connection weights of ANN to the investigated time series. The Mackey–Glass chaotic time series is first taken as a theoretical series to evaluate the efficiency, effectiveness, and robustness of the constructed EANN. Then we apply the EANN to the forecasting of 10-day reservoir inflows of the Shihmen Reservoir and compare the forecasted inflow with the lag-one autoregressive (AR(1)) and autoregressive moving-average with exogenous inputs (ARMAX) models.

The status of EANN

EANNs have been widely explored in last few years. Yao (1993, 1999) provided two excellent reviews of the different combinations between ANNs and EAs. He roughly divided the combinations into three evolutions of ANNs and then described a framework for EANNs and pointed out an important concept: "Design of the optimal architecture for an ANN can be formulated as a search problem in the architecture space where each point represents an architecture". There are several studies which have explored the applicability of EANN in the hydrological sciences. Cortez et al. (1996) used a genetic algorithm neural network (GANN) to forecast time series. Abraham (2004) proposed an effective evolutionary neural network, the meta-learning evolutionary artificial neural networks (MLEANN) and applied MLEANN to several different time series. Dawson et al. (2006) applied JavaSANE, a package developed from a symbiotic adaptive neuro-evolutionary (SANE) algorithm (Moriaty and Miikkulainen, 1998), to evolving and optimizing individual neurons of a rainfall–runoff network. Leahy et al. (2008) stated that a global optimization methodology for ANN architecture and weights can be employed successfully to a river level prediction. Chaves and Chang (2008) recently presented an Evolving ANN Intelligent System (ENNIS) for reservoir operation.

The parameters of ANN and GA, types and divisions of observed or generated data, and criteria of performance evaluation in the above studies and this study are tabulated in Table 1. The primary research questions to be addressed in this paper are as follows:

1. How can the architecture of ANN (e.g., inputs, the number of hidden layers, and the number of neurons in hidden layers) be automatically optimized?
2. How can the inputs be appropriately selected among all possible ones?
3. How can the parameters of ANN's architecture be encoded to artificial chromosomes?
4. How can the chromosomes be encoded so that they can be performed crossover with different lengths?
5. How well do EANNs perform in comparison to conventional ANNs in terms of effectiveness, efficiency, and robustness?

Modeling the EANN

Artificial neural networks

An ANN is a network system composed of a set of interconnected processing elements (neurons), with each element expressed by a function of the sum of weighted inputs as follows:

$$Y_i = f_i\left(\sum_i W_{ij}X_j - \theta_i\right) \tag{1}$$

where Y_i is the output of the ith neuron; f_i is the transfer function of the ith neuron; W_{ij} is the connection weight between the ith and jth neurons; X_j is the input of the jth neuron; θ_i is the threshold of the ith neuron.

The architecture of multi-layer feedforward BPNN can be roughly divided into three layers, including input layer, hidden layer, and output layer (Fig. 2). The input layer acts as the receiver of input data without a weighted sum. The hidden layer could be a single layer or multiple layers. In addition to receiving the information from all neurons of the input layer, each neuron of the hidden layers sums weighted inputs and then delivers to the neurons of next hidden layer or output layer. The output layer is a single layer including one or several output variables.

The estimated output value of the output layer is compared with a target output value by objective functions (or error functions). If the error does not meet the criteria of the objective functions, then the connection weights between any two feedforward-connected neurons are modified. To search optimal connection weights and biases, we may use the search algorithms to minimize the error functions. Common search algorithms include gradient descent method, Newton method, and conjugated gradient method.

Table 1
Comparison of three previous studies and this study applying EANNs.

Items	Cortez et al. (1996)	Abraham (2004)	Dawson et al. (2006)	This study
Types of ANN	Genetic algorithm neural network (GANN)	Meta-learning evolutionary artificial neural networks (MLEANN)	Symbiotic adaptive neuro-evolutionary (SANE) algorithm	Hybrid-encoding evolutionary artificial neural networks (HEEANN)
Types of data	91–349 generated data of time series	a. 475 observations of wastewater time series $f(t)$ b. 1000 generated data of Mackey–Glass chaotic time series $x(t)$ c. 292 pairs of methane $u(t)$ and carbon dioxide $y(t)$ time series	6-h streamflow and rainfall observations for 3 years	a. 1000 generated data of Mackey–Glass chaotic time series $x(t)$ b. 10-day reservoir inflow and rainfall observations for 39 years
Division of data	Training and validation	Training and testing	Training, validation, and testing	Training and testing
Input variables	3–14 neurons (unspecified variable)	a. $f(t-1)$, $f(t)$ b. $x(t-18)$, $x(t-12)$, $x(t-6)$, $x(t)$ c. $u(t)$, $y(t)$	Predefined 3 streamflow variables and 5 rainfall variables	a. Any selection of all variables between $x(t-18)$ and $x(t)$ b. Constant or any selection of variables between $P(t-2)$, $P(t-1)$, $P(t)$, $Q(t-2)$, $Q(t-1)$, and $Q(t)$ for observed and standardized data
Number of hidden layers and number of neurons in hidden layers	1 layer (3–14 neurons)	Maximum number of neurons = 16	Maximum number of neurons = 1000	Maximum number of hidden layers = 3; maximum number of neurons = 15
Neuron connection	Fully connected	N/A	N/A	Fully connected
Transfer function	N/A	Tanh, logisitic, sigmoidal, tanh–sigmoidal, log–sigmoidal	N/A	Tanh–sigmoidal and linear
Output variables	1-h ahead data	a. $f(t+1)$ b. $x(t+6)$ c. $y(t+1)$	6-h and 24-h ahead streamflow	a. $x(t+6)$ b. $Q(t+1)$
Learning rules	Backpropagation algorithm	Backpropagation algorithm, conjugate gradient algorithm Levenberg-Marquardt algorithm, quasi-Newton algorithm	Backpropagation algorithm	Scaled conjugate gradient algorithm
Encoding scheme	Binary indirect encoding	Binary direct encoding	N/A	Binary hybrid encoding (direct encoding and indirect encoding)
Encoded parameters	Number of inputs, transfer functions, learning rate, and number of neurons in hidden layer	Number of inputs, transfer functions, learning algorithms, number of hidden layers, number of neurons in hidden layers, and connection weights	Neuron	Inputs, number of neurons in hidden layer
Chromosome length	14 bits	N/A	N/A	a. 31 b. 18
Initial population	20	40	N/A	20
Genetic operators	Selection, crossover, and mutation – Roulette-wheel selection – Crossover rate = 1 – Mutation rate = 0.02	Selection and mutation – Rank selection – Elite rate = 0.05 – Mutation rate = 0.4	Selection, crossover, and mutation	Selection, crossover, and mutation – Elite number = 1 – Tournament selection – Crossover rate = 0.5 – Mutation rate = 0.05
Generations	N/A	40	200	10
Computing time	N/A	a. 288–1463 min b. 62–696 min c. 146–1176 min	N/A	a. 3–223 min b. 1.3–160 min
Iteration	N/A	N/A	Random 50 times	At least successive five times
Criteria of performance evaluation	MSE and SMSE	RMSE	RMSE, COE, MAE, and COD	RMSE and CC

Note: N/A means "not available".

Genetic algorithm

As briefly mentioned above, the GA, by mimicking the evolutionary process of natural genetic heredity, transfers the search process of an optimization problem into an evolutionary process. The parameters of an optimization problem are first encoded as an artificial chromosome. By starting with a number of randomly initialized chromosomes, the optimization process is then implemented for each chromosome and the fitness of each chromosome is evaluated by objective functions. If the optimal solution does not meet the criteria of the objective function or the predefined iteration times are not met, the optimization process is iterated through reproduction, crossover, and mutation until the stop criteria are satisfied. The key elements of the GA used in this study are briefly given as follows.

Encoding: Encoding is the first step of GA. All the parameters of a problem to be optimized have to be encoded as an artificial chromosome, i.e., a string of genes with fixed length.

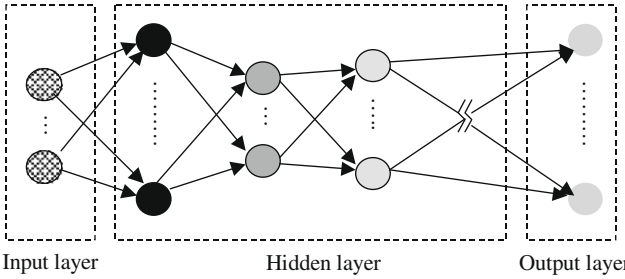


Fig. 2. Schematic architecture of feedforward BPNN.

Initialization: Randomness is the common way to initialize a number of individual chromosomes expressing the population.

Fitness: Fitness may be regarded as the degree of evaluation over the search process for the individual in every generation whether it is satisfied with the objective function.

Genetic operators

(1) Reproduction. Elite and selection of chromosomes are two strategies of reproduction. In elite strategy a chromosome with the best fitness is regarded as the elite among all per generation. All features of the elite chromosome will remain unchanged in the next generation. Selection means that some of the chromosomes, except for the elite, with better fitness will be put into the match pool awaiting the opportunity of crossover.

(2) Crossover. Increasing diversity of genes is the purpose of crossover. The way of crossover is to randomly select any two of all parent individuals in the match pool, then exchange some genes of each parent individual.

(3) Mutation. Mutation can avoid the optimal search trapped in the local minimums of error function. A typical way of gene mutation is to randomly select some genes of parent individuals, and then change the values of the selected genes into different ones.

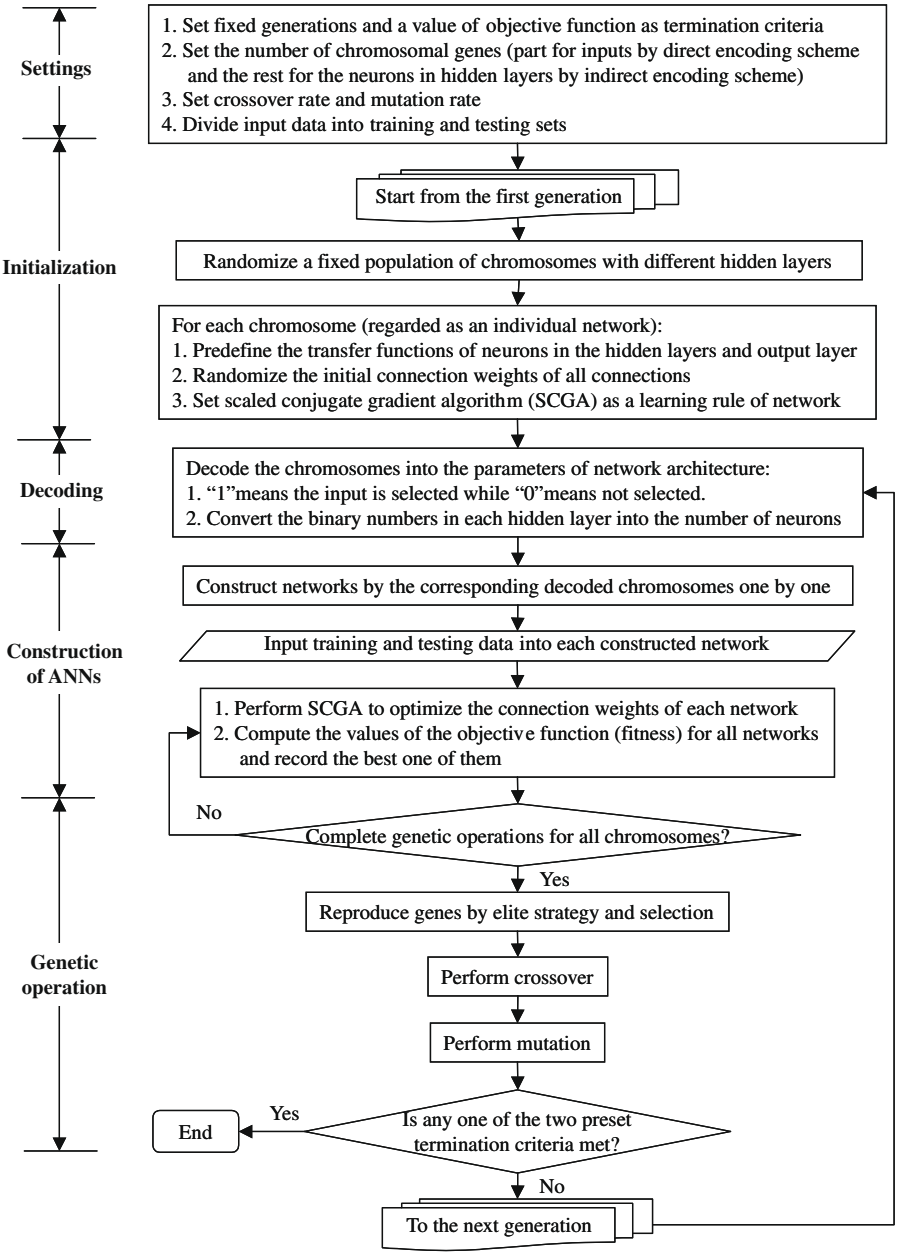


Fig. 3. Flowchart of the proposed EANN modeling.

Termination of genetic search

The evolutionary process of optimization search can be iterated unless the termination criteria have been met.

Evolutionary artificial neural networks

This study is to evolve the network architecture of BPNN by GA and optimize the connection weights by a gradient search. The detailed steps of our framework are described as follows and shown in Fig. 3.

Encoding of architecture

The encoding of ANN's architecture can be roughly classified into direct encoding scheme and indirect encoding scheme as follows:

Direct encoding scheme:

The way to encode the connection of any two among all neurons of an ANN is called direct encoding. In terms of network architecture, the connection between any two neurons is generally expressed by a binary digit: 1 means connection and 0 no connection. Having complete information is the advantage of a direct encoding scheme. However, if the network architecture becomes more complicated, the chromosome length or connectivity matrix would significantly increase and then cause inefficiency. Therefore, direct encoding is more suitable for a small network.

Indirect encoding scheme:

An indirect encoding scheme involves only encoding some characteristics of the network architecture such as the number of hidden layers or that of neurons. The search process would be more efficient, however, if some important parameters of the network architecture were pre-selected and encoded (Abraham, 2004).

Hybridized encoding scheme:

A hybridized encoding scheme, a combination of direct and indirect encoding, is proposed in this study. The network is composed of an input layer, multiple hidden layers, and an output layer; and the network is fully connected. To automatically identify appropriate inputs, a direct encoding scheme is used to encode the input vector of the network, the first part of the chromosome. Indirect encoding is used to encode the number of neurons in hidden layers, the rest part of the chromosome. Fig. 4 shows a schematic encoding of the network architecture for a fully connected feedforward ANN. Fig. 5 illustrates an example of a feedforward ANN's architecture corresponding to its chromosome.

The details of the hybridized encoding schemes are described as follows:

1. Direct encoding the inputs:
   (1) Predefine the number of possible inputs (no_input).
   (2) Randomly generate a series of 0 or 1, where "1" means the input is selected while "0" means not selected, for all the possible inputs.
2. Indirect encoding the number of neurons in hidden layers:
   (1) Predefine a maximal number of hidden layers (no_maxlayer) and a maximal number of neurons (no_gene) in hidden layers. It implies that the maximal string of genes from indirect encoded the neurons of hidden layers is no_maxlayer$*$ no_gene.
   (2) Randomly generate a number for the no_layer, which should not be larger than the number of predefined maximal hidden layers. The generated number (no_layer) represents the number of hidden layers for an individual chromosome.
   (3) Randomly generate a series of 0 or 1 for the no_gene of each layer. The total generated amount of the binary numbers is no_layer$*$no_gene.
   (4) Decode the binary numbers in each hidden layer into the number of neurons in the hidden layer by the following expression:

$$f(1)\cdot 2^{\text{no\_gene}} + f(2)\cdot 2^{\text{no\_gene}-1} + \ldots + f(\text{no\_gene})\cdot 2^{0}$$

   where $f(i) = 0$ or 1 ($i = 1,2,\ldots,$no_gene).
   (5) If all the generated numbers of neurons in different hidden layers are zeros, steps 1–3 are repeated until at least one gene is not zero.

Initialization of architectural chromosomes and network neurons

A number of network architectural chromosomes (no_indvidual) are initially randomized. It should be noted that the length of each chromosome would not be the same because of the randomly generated number of inputs and hidden layers. Each randomized chromosome is regarded as an individual network. For each network, the transfer functions of neurons in the hidden layers and output layer are defined as tan–sigmoid transfer functions and linear transfer function, respectively. The initial connection weights for each constructed network are also randomized.

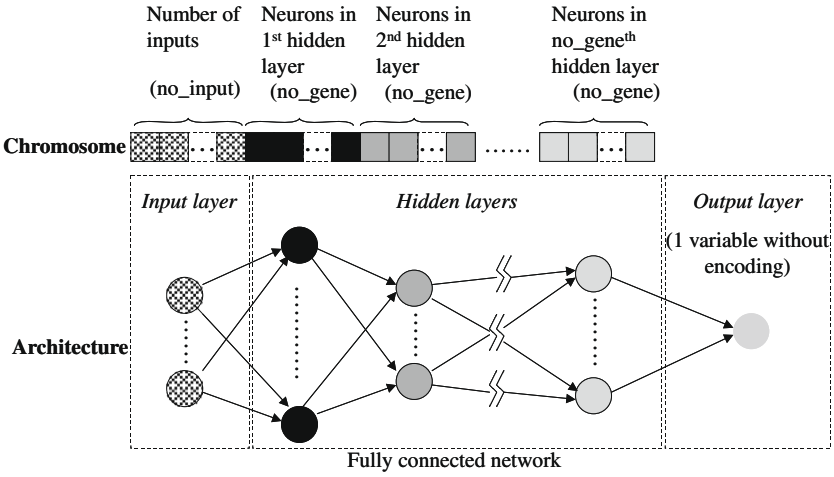


Fig. 4. Schematic encoding of ANN's architecture.

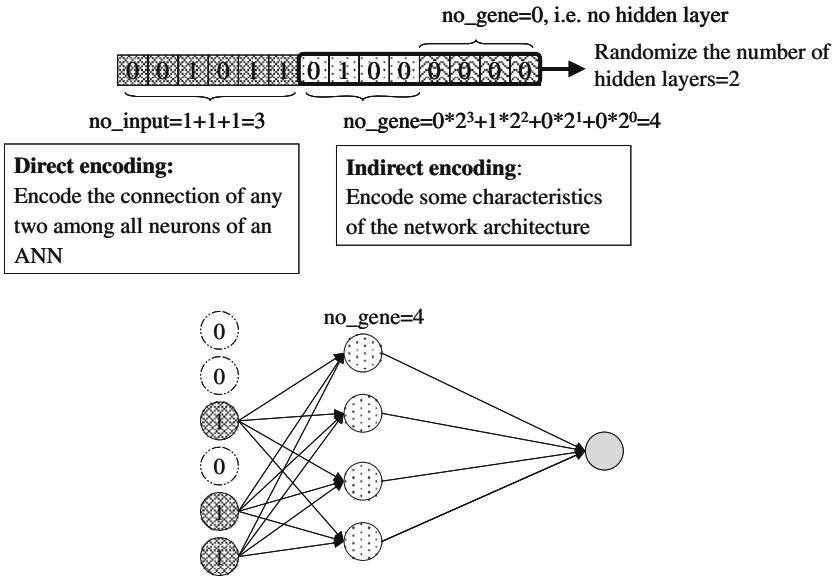


Fig. 5. An example of a feedforward ANN's architecture corresponding to its chromosome.

Construction of a feedforward ANN

1. Construct feedforward ANNs by the corresponding decoded chromosomes one by one. The network architecture is determined by the following rules:
   (1) If the number of hidden layers is randomized as one (no_layer = 1) and the number of neurons in the layer is not zero, there is only a hidden layer in the network.
   (2) If no_layer = 2 and the numbers of neurons in both layers are not zero, there are two hidden layers in the network. However, if either one of the two numbers is zero, there is only one hidden layer in the network.
   (3) The above rules are applied when the architecture of the network for the randomized number of hidden layers is more than two.
2. Use the scaled conjugate gradient algorithm (SCGA) (Moller, 1993) as a learning rule to search for the optimal connecting weights of the constructed ANN. SCGA is a supervised learning algorithm with superlinear convergence rate and based upon a class of conjugate gradient (CG) methods, which are well-known numerical techniques used for solving various optimization problems (Ham and Kostanic, 2001). In practice, the process of CG makes good uniform progress toward the solution at every time step and has been found to be effective in finding better optimization than the standard BP algorithm (Chiang et al., 2004).
3. Set an objective function (e.g., minimal error between target and forecasted outputs of a network) and search the optimal connecting weights for each constructed network by the SCGA using the training set, then compute the value of the objective function using the testing set.

Genetic operations

1. Set the number of generations (no_generation) and then perform the following genetic processes generation by generation.
2. Compute the values of the objective function for all chromosomes in the current generation and record the best one. Each of the objective functions could be regarded as the fitness of the specific chromosome.
3. Perform genetic operations for each generation. Parent chromosomes with better fitness have more probabilities to generate offspring chromosomes. The genetic operators include reproduction, crossover, and mutation, described as follows:
   (1) Reproduction: First, one or a few with better fitness (no_elite) from parent chromosomes in the current generation, called elite chromosomes, are selected and kept unchanged in the next generations. Then two that are not elite chromosomes are randomly selected by tournament selection and one with better fitness is put into a match pool.
   (2) Crossover: Set a crossover rate (prob_crossover) and define the number of crossovers (no_crossover, should be an integer) as follows:

$$\text{no\_crossover} = \text{prob\_crossover} * (\text{no\_individual-no\_elite}) \tag{2}$$

   Repeatedly select chromosomes for crossover in the match pool for several times (no_crossover/2). To perform crossover, two chromosomes exchange their post-parts. That is, one offspring chromosome is combined with a pre-part of the first parent chromosome and a post-part of the second chromosome, whereas the other offspring chromosome is combined with a pre-part of the second parent chromosome and a post-part of the first chromosome. Fig. 6 illustrates a schematic crossover for two chromosomes.
   (3) Mutation: Set a mutation rate (prob_mutation) and define the number of genes to be mutated (no_mutation, also an integer) as follows:

$$\text{no\_mutation} = \text{prob\_mutation} * (\text{no\_individual-no\_elite}) * (\text{no\_input} + \text{no\_layer}) \tag{3}$$

   Repeatedly select chromosomes for mutation except elites. One parent chromosome is randomly selected at a time. Then a gene of the chromosome is randomly selected for mutation, changing into a different binary number.
   (4) Repeat step 2 for offspring chromosomes in the next generations until the termination criterion has been met.
   (5) Iterate all steps for a number of times.

Applications

Mackey–Glass chaotic time series

Mackey and Glass (1977) proposed the following first-order differential-delay equation:

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \quad (4)$$

The time data generated by Eq. (4) are well known as the Mackey–Glass chaotic time series, being periodic when τ is small and chaotic when $\tau > 17$. Mackey–Glass time series are usually used as a time series benchmark when investigating performance of artificial neural network.

To evaluate the performance of the proposed EANN, this study uses Mackey–Glass time series available from the data file of MATLAB®, a widely used computer platform. The 1200 used data have a mean of 0.9194 and a standard deviation of 0.2357. To reduce the effect of noise, the first 118 and the last 82 data of the time series were not used. Eight hundred of the 1000 data available are used as training set and the rest two hundreds as testing set. It should be noted here that the validation set is not necessary for EANN modeling. Dawson et al. (2006) pointed out that the purpose of constructing one model on one dataset and using a second dataset for conventional ANNs is to prevent an over-fitted solution; while, for GA-based ANNs, constructing several models on one dataset and using a second dataset is to select the best available model from what might be a set of over-fitted solutions. To further explore this issue and to establish the degree to which the evolved models are capable of being overfitted would require the second dataset to be used directly in the neuro-evolutionary process. Abraham (2004) used the same way of dividing the data sets into training and testing sets for his meta-learning EANN.

Parameter settings

The parameters of EANN are described as follows:

1. Most of previous studies used the four input variables, $x(t-18)$, $x(t-12)$, $x(t-6)$, $x(t)$, to estimate the single output variable, $x(t+6)$. Since direct encoding allows all possible variables as inputs to ANN, this study uses all variables between $x(t-18)$ and $x(t)$ as input variables (i.e., $x(t-\Delta t)$, where $\Delta t = 0,1,2,\ldots,18$) and $x(t+6)$ as output variables.
2. The maximal hidden layers are set to be 3 and the number of neuron genes in each layer is 4 (i.e., the neuron range can be 0–15). The total number of neuron genes in three hidden layers is not larger than 12 (i.e., 3∗4).
3. The population of chromosomes is initialized as 20, the same as the study of Cortez et al. (1996).
4. The number of elites is set to be 1.
5. The crossover rate and mutation rate are set to be 0.5 and 0.05, respectively.
6. The generations are set to be 10.
7. SCGA is used as learning rule of ANN and, to make sure the search process could research optimal solutions, the training epoch is set as 8000.
8. The root mean square error (RMSE), expressed in Eq. (5), is used as the termination criterion. Three different RMSEs (0.001, 0.0005, and 0.0004, respectively) are set, where 0.0004 was the smallest RMSE value used in Abraham's (2004) (see Table 2).

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{t=1}^{N}(x_p(t+6) - x_o(t+6))^2}{N}} \quad (5)$$

in which $x_p(t+6)$ is the forecasting value at $t+6$; $x_o(t+6)$ is the generated data at $t+6$; N is the number of generated data.

9. To avoid over-fitting, the maximal connections of the constructed network architecture are restricted to be less than 200.

Results and discussion

1. Table 3 shows the results of three arbitrary runs for EANN modeling, under three different termination criteria, (a) RMSE $\leqq$ 0.001, (b) RMSE $\leqq$ 0.0005, and (c) RMSE $\leqq$ 0.0004 on testing data set, respectively. It appears the EANN can effectively obtain suitable networks. With the same termination criterion, several other runs have been done and various network architectures are obtained. The results indicate that the inputs are different and the number of the hidden layers could be one, two, or three. The difference is mainly due to the initial randomness and genetic operations in EANN modeling. It should be noted that the initial random selection of inputs might leave out important inputs and prior information would be helpful to determine the potential inputs at the phase of settings. In this study, we aim at automatically identifying the optimal architecture of ANN and a large number of input combinations have been generated and evaluated through GA process, and the initially selected inputs without contributions to meet the objective function in the first generation would be changed into more important inputs after generations of genetic operation.

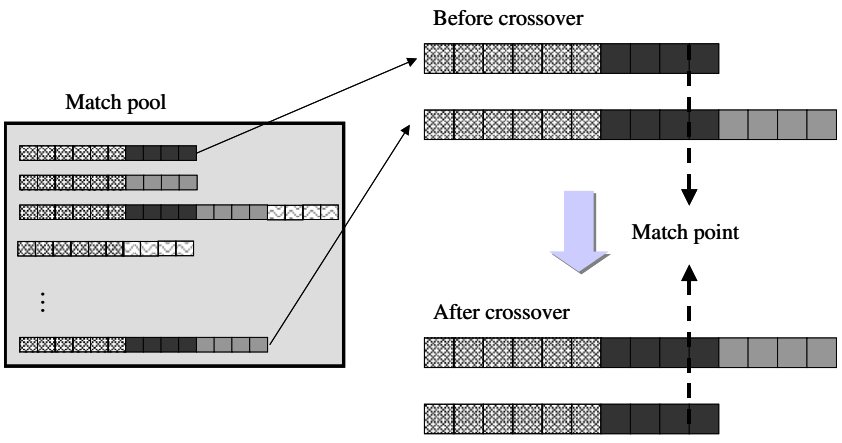


Fig. 6. Schematic crossover for two chromosomes.

Table 2
Parameter settings of EANN modeling for Mackey–Glass time series.

Number of input genes (no_input)	19	Maximal hidden layers (no_maxlayer)	3
Number of neurons [gene] in each hidden layer (no_gene)	15 [4]	Initial population of individual chromosomes (no_individual)	20
Number of elite chromosomes (no_elite)	1	Crossover rate (prob_crossover)	0.5
Training epochs	8000	Mutation rate (prob_mutation)	0.05
Learning rule	Scaled conjugate gradient algorithm (SCGA)		
Termination criterion	(1) Generations (no_generation)		10
	(2) RMSE value (training)		(a) RMSE $\leqq$ 0.0010
			(b) RMSE $\leqq$ 0.0005
			(c) RMSE $\leqq$ 0.0004

Table 3
Results of three arbitrary runs for EANN modeling under different termination criteria.

	Termination criterion (a) RMSE $\leqq$ 0.001	Termination criterion (b) RMSE $\leqq$ 0.0005	Termination criterion (c) RMSE $\leqq$ 0.0004
The generation when termination criterion is met	1st	2nd	2nd
Optimal inputs	18, 16, 15, 14, 12, 11, 6, 3, 2, 0	18, 17, 16, 13, 12, 11, 9, 8, 7, 6, 5, 4	17, 16, 15, 13, 12, 11, 10, 9, 7, 5, 3, 2
Optimal number of neurons in hidden layers	14	3-8-9	10
RMSE (training)	0.0009998	0.0004983	0.0003998
RMSE (testing)	0.0009306	0.0005505	0.0004266

Note:
1. In each run, EANNs are modeled for ten generations. The modeling is terminated once the termination criterion is met.
2. The number in the third row, e.g., "18", means $x(t-18)$.
3. The numbers in the fourth row, e.g., "3-8-9", mean there are three hidden layers and the number of neurons in the first hidden layer is 3, in the second is 8, and in the third is 9.

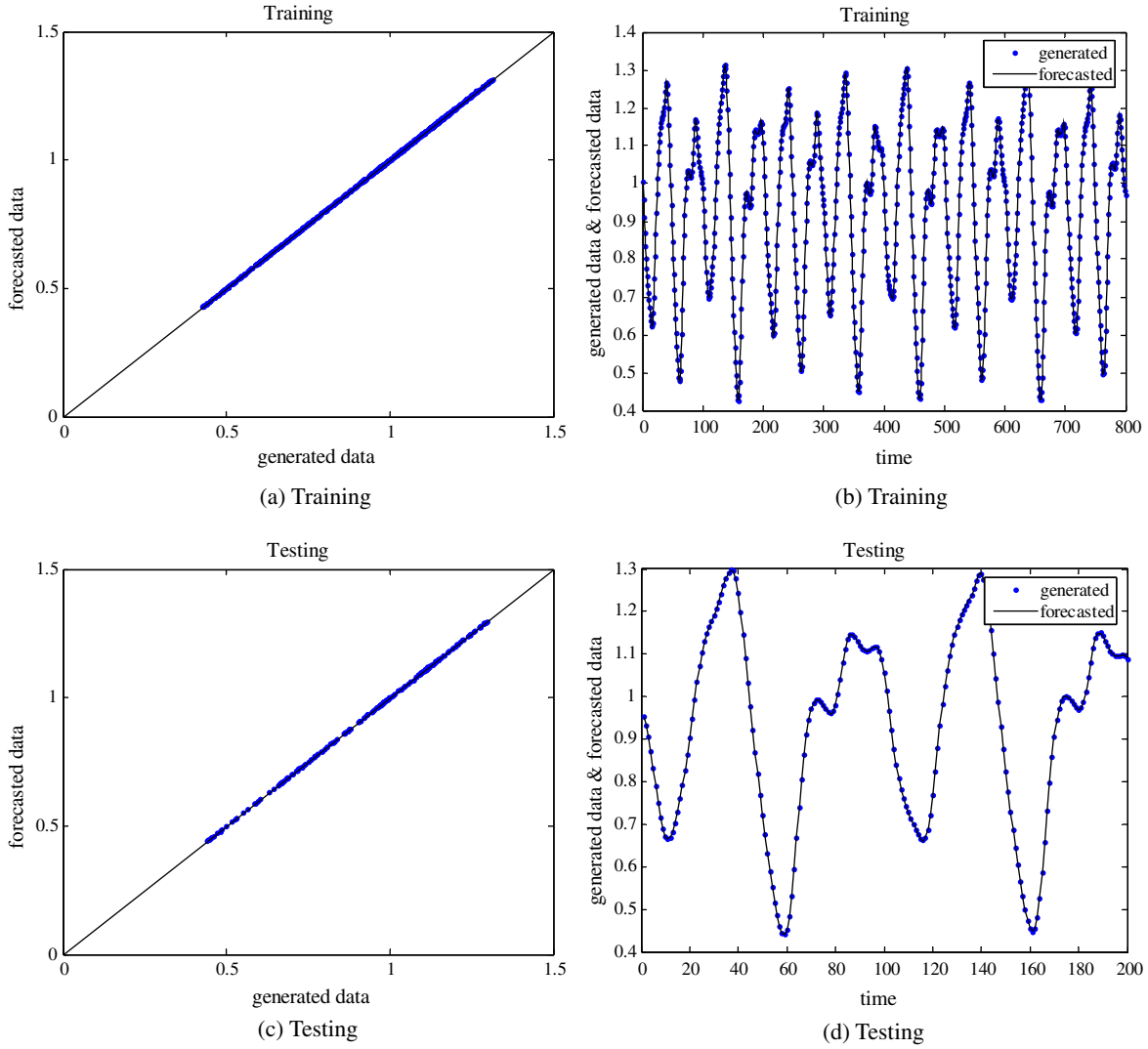


Fig. 7. Comparison between forecasted and generated time series for the arbitrary run under the termination criterion RMSE $\leqq$ 0.0004 in Table 3.

Table 4
The computation time (in minutes) of five arbitrary runs for EANN modeling under different termination criteria.

Successive run	Termination criterion (a) RMSE ≦ 0.001	Termination criterion (b) RMSE ≦ 0.0005	Termination criterion (c) RMSE ≦ 0.0004
Computing time (min.)			
1st	8	87	57
2nd	6	180	223
3rd	10	57	184
4th	8	110	77
5th	3	67	161

2. Another important point to be addressed is whether the optimal architecture should be unique or consistent. It can be found from the comparison of two results: The forecasting results of RMSE ≦ 0.0004 are compared with those of Abraham's study (2004), whose termination criterion of RMSE is also 0.0004. Under such a strict requirement of accuracy, it is however found that there are still several optimal results, i.e., multi-optimal solutions with different network architectures. A stricter termination criterion might be helpful for a more consistent result; but it could not be assured of finding a unique optimal solution unless we have known the exact global optimal solution.
3. There are chances that the initial population happens to include a set of solutions close to the "stopping criterion". For the termination criteria RMSE ≦ 0.001 in Table 3, the optimal results are obtained in the first generation. In addition to the chance of the initial population close to the termination criterion, it is also because of powerful searching ability of SCGA.
4. Fig. 7 displays a comparison between the forecasted and generated time series for the arbitrary run under the termination criterion (c) in Table 3. It appears that almost all pairs of forecasted and generated data points are on the diagonal line, which means nearly perfect forecasting has been achieved. Therefore, the EANN can be regarded as having the ability to forecast with higher accuracy.
5. In addition to effectiveness, the other two criteria used to evaluate the performance are robustness and efficiency. Robustness is the ability to find an optimum for every search under restriction, while efficiency is the ability to take the least time to search for the optimum. Table 4 shows the results of five arbitrary runs for EANN under three different termination criteria. It appears optimal network architecture can be found for every run under its restriction, which means the proposed EANN can be regarded as robust. It can also be seen that the time to search for optimal network architectures ranges from 3 to 223 min for three termination criteria (based on a laptop with Intel Pentium M, 1.73 GHz CPU and 512 MB RAM). Compared with the computing time in Abraham's (2004) study, the proposed EANN could be regarded as efficient, in spite of the different evolutionary computation and computer.
6. The parameters of GA inevitably require a number of trials to increase the possibility of searching the global optimal solution. The trials of parameters usually rely on the features of data. In this study, the settings of the crossover rate and mutation rate are determined after a number of trials. The maximum number of hidden layers and maximum number of neuron in the hidden layer might increase the effectiveness of the EANN model. However, they would also increase the possibility of over-fitting and training time. With respect to the parameter of SCGA, we also noticed that increase of training epochs to a certain number (e.g., 8000 epochs in the case of chaotic series) would improve the performance.

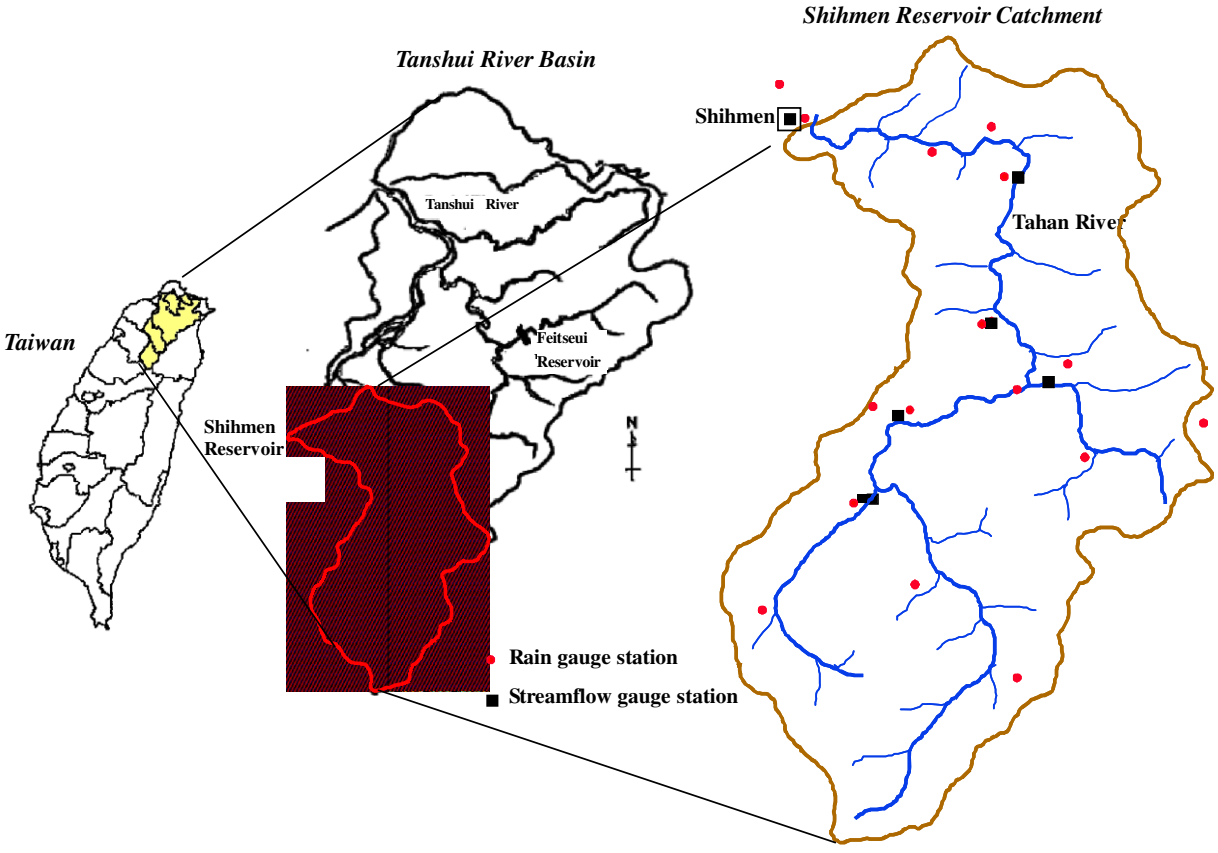


Fig. 8. Locations of study area and gauge stations.

Table 5
Parameter settings of EANN modeling for reservoir inflow time series.

Number of input genes (no_input)	6	Maximal hidden layers (no_maxlayer)	3
Number of neurons [gene] in each hidden layer (no_gene)	15 [4]	Initial population of individual chromosomes (no_individual)	20
Number of elite chromosomes (no_elite)	1	Crossover rate (prob_crossover)	0.5
Training epochs	5000	Mutation rate (prob_mutation)	0.05
Learning rule	Scaled conjugate gradient algorithm (SCGA)		
Termination criterion	(1) Generations (no_generation)		10
	(2) Initial value of RMSE (testing)		RMSE $\leqq$ 60

Reservoir inflow time series

Description of study area

Streamflow forecasting is of significant importance for planning and operation of water resource systems. For hydrologic components, there is a need for short-term (hourly or daily), mid-term (10-days or monthly), and long-term (yearly) forecasts of streamflow events in order to optimize the systems or to plan for future expansion or reduction. Mid-term streamflow forecasting is especially important for the operation of water supply systems over drought seasons. Among the water supply systems, reservoirs should be regarded as the most important and effective water storage facilities, which have the functions of modifying uneven distribution of water and allocating water resources. Precise forecasting of seasonal inflows of reservoirs will benefit the reservoir operation and management.

The Shihmen Reservoir is located in the upper reaches of the Tahan River, a branch of Tanshui River in northern Taiwan. The watershed area of the reservoir is 763.4 km^2 and the effective water storage is 251.88 million cubic meters. The reservoir serves a number of purposes, including irrigation, hydroelectric power, fresh water supply, flood prevention and sightseeing. Supplying water to 28 districts and housing to 3.4 million people, the reservoir is a very important water facility for the livelihoods of the people living in northern Taiwan.

Table 6
Performance of AR(1) and ARMAX models.

Models	Forecasting	RMSE (training)	RMSE (testing)
AR(1)	$Q(t+1)$	47.2	58.9
ARMAX(1, 2, 1)	$Q(t+1)$	46.8	59.2

Note:
1. AR(1) model: $Q(t+1) = 0.6186 \cdot Q(t) + e(t+1)$, where $e(t+1)$ is an error term at time $t+1$.
2. ARMAX(1, 2, 1) model: $Q(t+1) = 0.7257 \cdot Q(t) + 0.1866 \cdot P(t) - 0.06515 \cdot P(t-1) + e(t+1) - 0.2239 \cdot e(t)$.
3. The time series for input were standardized before modeling for both AR(1) and ARMAX models.
4. A time step is defined as ten days, so "$t+1$" means ten days ahead.

Locations of the study area and gauge stations are shown in Fig. 8. The weighted average rainfalls over the watershed are computed by Thiessen method and the reservoir inflow measurements with time step of ten days are available from the gauging stations. The data were collected from the gauging stations during the period from 1965 to 2003 and divided into a training set, 30 years, and testing set, 9 years. The total numbers of data sets are 1080 ten-days for training and 324 ten-days for testing, respectively.

Parameter settings

The parameters of the proposed EANN are described as follows and tabulated in Table 5:

1. In order to compare with the performance of AR(1) and ARMAX models, which are usually applied to periodical streamflow forecasting, two sets of rainfall and inflow variables are taken as the inputs of EANN. One set only includes inflow variables, $Q(t)$. The other set includes both rainfall and inflow variables, $P(t-\Delta t)$ and $Q(t-\Delta t)$ ($\Delta t = 0, 1, \ldots, \Delta t_{\max}$), where the time step is 10 days and the value of $\Delta t_{\max}$ is arbitrarily set to be two. The output variables of both sets are $Q(t+1)$. The value of $\Delta t_{\max}$ could be either set arbitrary or based on prior information of inputs.
2. SCGA is again used as a learning rule of ANN and the training epochs are set to be 5000.
3. The termination criterion of EANN can be set by an RMSE value. Similarly, to compare with the performance of AR(1) and ARMAX models in Table 6, the RMSE values on a testing data set are initially set to be 60 and then decreased or increased if an optimal result is found or not found, respectively. In addition to RMSE, Eqs. (6) and (7) are also used to estimate the correlation and mean relative error between the forecasted and observed inflow, respectively.

$$r = \frac{\sum_{t=1}^{N}[Q_f(t+1) - \bar{Q}_f(t+1)][Q_o(t+1) - \bar{Q}_o(t+1)]}{\sqrt{\sum_{t=1}^{N}[Q_f(t+1) - \bar{Q}_f(t+1)]^2}\sqrt{\sum_{t=1}^{N}[Q_o(t+1) - \bar{Q}_o(t+1)]^2}} \quad (6)$$

$$\mathrm{MRE} = \frac{\frac{1}{N}\sum_{t=1}^{N}(Q_f(t+1) - Q_o(t+1))}{\overline{Q_o(t+1)}} \quad (7)$$

Table 7
Cases settings with different inputs.

	Inputs	Forecasting	Initial termination criterion
Case 1	Random selection of $P(t)$, $P(t-1)$, $P(t-2)$, $Q(t)$, $Q(t-1)$, $Q(t-2)$	$Q(t+1)$	RMSE $\leqq$ 60
Case 2	$P(t)$, $P(t-1)$, $Q(t)$	$Q(t+1)$	RMSE $\leqq$ 60
Case 3	$Q(t)$	$Q(t+1)$	RMSE $\leqq$ 60
Case 4	Random selection of $P^S(t)$, $P^S(t-1)$, $P^S(t-2)$, $Q^S(t)$, $Q^S(t-1)$, $Q^S(t-2)$	$Q(t+1)$	RMSE $\leqq$ 60
Case 5	$P^S(t)$, $P^S(t-1)$, $Q^S(t)$	$Q(t+1)$	RMSE $\leqq$ 60
Case 6	$Q^S(t)$	$Q(t+1)$	RMSE $\leqq$ 60

Note:
1. P and Q mean observed rainfall and inflow, respectively. The superscript S means standardization, i.e., P^S and Q^S mean standardized rainfall and inflow, respectively.
2. Time step is defined as ten days.
3. The RMSE values as termination criterion could be decreased or increased if the optimal result is found or not found, respectively.

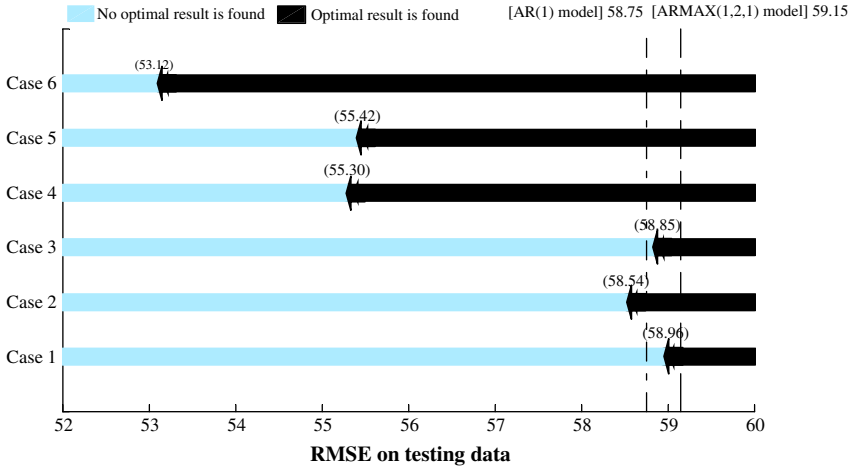


Fig. 9. Ranges of optimal RMSE values on testing data for six cases.

Table 8
The best optimal results of the EANN modeling in six cases.

		Case 1	Case 2	Case 3	Case 4	Case 5	Case 6
The generation when termination criterion is met		2	3	1	1	4	7
Optimal inputs		P: 1 Q: 0	P: 0, 1 Q: 0	Q: 0	P^S: 1 Q^S: 1	P^S: 0, 1 Q^S: 0	Q^S: 0
Optimal number of neurons in hidden layers		2-13	2-15	1-6	3-3	10-4-11	7-12-9
RMSE	Training	44.9	45.2	46.4	43.6	41.5	43.5
	Testing	58.9	58.5	58.8	55.3	55.4	53.1
CC	Training	0.57	0.56	0.52	0.60	0.65	0.60
	Testing	0.52	0.54	0.52	0.59	0.62	0.63
MRE	Training	0.002	−0.005	0.015	0.018	0.012	0.010
	Testing	0.122	0.056	0.041	0.062	0.088	0.030

Note:
1. Q and P in the 3rd row denote inflows and rainfalls, respectively. For example, "P: 0, 1" means the rainfall variables $P(t)$ and $P(t-1)$, while "Q: 0" means the inflow variables $Q(t)$. Q^S and P^S in the same row denote standardized inflows and rainfalls, respectively.
2."RMSE" in the 5th and 6th rows means "root mean square error", "CC" in the 7th and 8th rows means "coefficient of correlation", and "MRE" in the 7th and 8th rows means "mean relative error".

where $Q_f(t+1)$ is the forecasted inflow at time $t+1$; $\overline{Q_f}(t+1)$, the average of $Q_f(t+1)$ over N; $Q_p(t+1)$, the observed inflow at time $t+1$; and $\overline{Q_o}(t+1)$ is the average of $Q_p(t+1)$ over N; N is the number of data.

Case settings

To investigate the performance of the proposed EANN, we designed six different cases. Table 7 displays the case settings with different inputs and the same forecasting and initial termination criteria. The six cases can be divided into two groups: the inputs with observed data (cases 1, 2, and 3) and those with standardized data (cases 4, 5, and 6). Since standardization of seasonal time series data is usually used to eliminate the effects of periodical trends, the purpose of the second group is to investigate whether standardization could improve the performance of the proposed EANN.

The standardization of rainfall and inflow data can be attained by Eqs. (8) and (9), respectively.

$$P_{ij}^S(t) = \frac{P_{ij}(t) - \overline{P_j(t)}}{\sigma_{P_j}(t)} \tag{8}$$

$$Q_{ij}^S(t) = \frac{Q_{ij}(t) - \overline{Q_j(t)}}{\sigma_{Q_j}(t)} \tag{9}$$

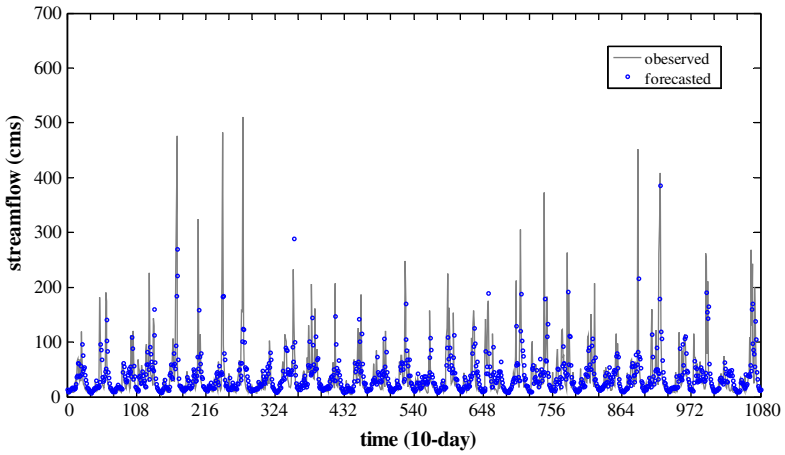
where $P_{ij}^S(t)$ is the standardized rainfall of the jth 10-days in ith year; $P_{ij}(t)$, the observed rainfall of the jth 10-days in ith year; $\overline{P_j(t)}$, the average of $P_{ij}(t)$; $\sigma_{P_j}(t)$, the standard deviation of $P_{ij}(t)$; $Q_{ij}^S(t)$, the standardized inflow of the jth 10-days; $Q_{ij}(t)$, the observed inflow of the jth 10-days in ith year; $\overline{Q_j(t)}$, the average of $Q_{ij}(t)$; and $\sigma_{Q_j}(t)$ is the standard deviation of $Q_{ij}(t)$.

Results and discussion

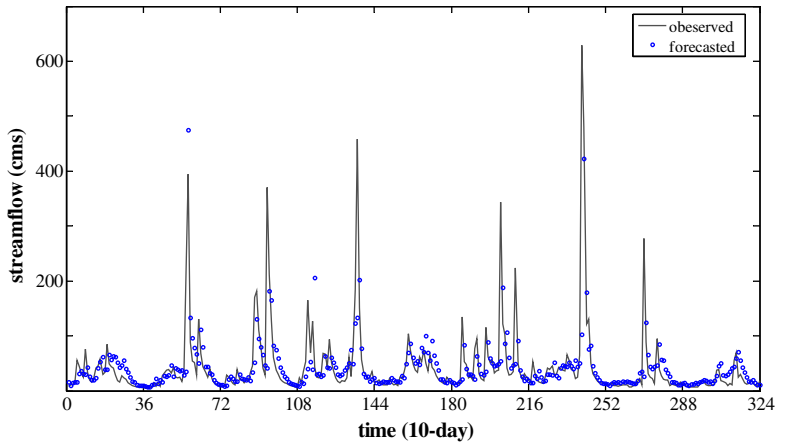
1. Fig. 9 shows the ranges (black bar) of optimal RMSE values on testing data for six cases. From the figure it can be also observed that the RMSE value on testing data in every case is smaller than that of ARMAX(1, 2, 1) model with standardized inputs, 59.15, and the RMSE values in cases 2, 4, 5, and 6 are smaller than that of AR(1) model with standardized inputs, 58.75. However, the RMSE values in cases 4, 5, and 6 are much smaller than those in cases 1, 2, and 3. The result reveals that standardization of input data can improve the performance of the proposed EANN. This could be because the standardized streamflow used as inputs provides the flow anomaly (e.g., below or above average flow) and supports information relevant to the dry or wet in climate scenario; as a result, they give better 10-day forecasts than that of non- standardized flow as inputs.
2. Table 8 shows the best optimal results of the EANN modeling in six cases. For example, in case 6 the termination criterion is met in the seventh generation; the input is only the standardized inflow at time t; there are three optimal hidden layers and

Table 9
Percentage of improvement for the RMSE values on training and testing data of EANN modeling, compared with ARMAX(1, 2, 1) and AR(1) models.

Models	ARMAX(1, 2, 1)	EANN		AR(1)	EANN
		Case 4	Case 5		Case 6
RMSE on training data	46.8	43.6	41.5	47.2	43.5
Percentage of improvement	–	6.8%	11.3%	–	7.8%
RMSE on testing data	59.2	55.3	55.4	58.9	53.1
Percentage of improvement	–	6.6%	6.4%	–	9.9%



(a) Training



(b) Testing

Fig. 10. Comparison between forecasted and observed inflow time series for case 6.

the number of neurons in the first hidden layer is 7, in the second is 12, and in the third is 9, respectively. The optimal coefficients of correlation on the training and testing data sets are 0.596 and 0.633, respectively. The optimal mean relative errors (MRE) are very small on both training and testing data sets (i.e., 0.01 and 0.03). Fig. 10 (a) and (b) display the comparison between forecasted and observed inflow time series for case 6 in Table 8. Since the performance of forecasting is our focus, we pay more attention to the results of testing. From Fig. 10 (b) it can be seen the proposed EANN performs well on the testing data set in the forecasting of low inflows. However, most of high inflow forecasting, especially the streamflow more than 150 m^3/s, shows underestimation. This is mainly because there are only a few typhoon events with limited observed high flow data to model and train the constructed networks.

3. The results in Table 8 also show that the case 5 (including the rainfall data) could have better performance than the case 6 (excluding the rainfall data) in training phase, however, a reverse result is obtained in the testing phase. Since the quick response from heavy precipitation can be in a matter of few hours for the catchment of size 763.4 km^2, we believe the rainfall information in this study cases could be regarded as disturbance in forecasting the 10-day ahead reservoir inflow. It can also been seen from that the RMSE of AR(1) is smaller than that of ARMAX(1, 2, 1).
4. The percentage of improvement for the RMSE values on training and testing data of EANN, compared with ARMAX(1, 2, 1) and AR(1) models with standardized inputs, is shown Table 9. Since the cases with standardized input data have better performance, cases 4, 5, and 6 are selected to compare with the

above two models. Compared with the ARMAX(1, 2, 1) model, the percentage of improvement in terms of the RMSE values on training and testing data for case 4 is 6.8% and 6.6%; for case 5 is 11.3% and 6.4%, respectively. Compared with the AR(1) model, the percentages of improvement in terms of the RMSE on training and testing data for case 6 are 7.8% and 9.9%, respectively, which is quite valuable and uneasy accomplishment for long-term hydrological time series forecast. The results reveal that the admirable effectiveness of the proposed EANN and standardization is beneficial to modeling for seasonal time series.

Conclusions

To pursue adaptivity and to increase the efficiency of optimization systems, there has been an increasing interest in a new general framework for adaptive systems, namely, Evolutionary Artificial Neural Networks, where the modeling potentialities of artificial neural networks have been matched with the adaptation properties of the evolutionary algorithms. The need for adaptation came out from several real-world applications in non-stationary environments such as non-linear control tasks and time series forecasting. This paper proposes a novel evolutionary neural network for hydrological time series forecasting. The excellent performance for forecasting of the Mackey–Glass chaotic time series shows that the proposed EANN concurrently possesses efficiency, effectiveness, and robustness. Furthermore, the forecasting of 10-day reservoir inflows reveals again the excellent effectiveness of the proposed EANN, and standardization is beneficial to modeling for seasonal time series.

The proposed EANN in this study consists of the following features:

1. The optimal architecture of feedforward ANN, including inputs, hidden layers, and neurons in each hidden layer, can be automatically searched. The automatic search algorithm has improved on the drawbacks of the conventional approach that requires predefining the network architecture and involves a tedious trail-and error process.
2. Binary direct encoding and indirect encoding are hybridized to encode the important parameters of network architecture into an artificial chromosome. One part of the chromosome, genes of inputs, is encoded by a direct encoding scheme, while the other part, neurons in hidden layers, is encoded by an indirect encoding scheme. A relatively short length of chromosome and simple encoding scheme are the advantages of such hybrid encoding schemes. The proposed EANN that allows evolving all possible input variables is better than the conventional evolutionary design of ANN, which only allows evolving the number of inputs or unchanged inputs.
3. Since the number of hidden layers is randomly initialized, the lengths of chromosomes in each generation cannot be the same. After crossover between two parent chromosomes with different lengths, the lengths of offspring chromosomes are also different. In comparison with some conventional evolutionary designs of ANN which only allow mutation, the proposed EANN can perform crossover with non-constant lengths of chromosomes in addition to mutation.

Acknowledgments

This study was partially supported by the National Science Council, ROC (Grant No. NSC 97-2313-B-002-013-MY3). In addition, the authors are indebted to the Editors and Reviewers for their valuable comments and suggestions.

References

Abraham, A., 2004. Meta learning evolutionary artificial neural networks. Neurocomputing 56, 1–38.
Abrahart, R.J., See, L., Kneale P.E., 1998. New tools for neurohydrologist: using network pruning and model breeding algorithm to discover optimum inputs and architecture. In: Proceedings of the 3rd International Conference on Geocomputation. University of Bristol.
Angeline, P.J., Saunders, G.B., Pollack, J.B., 1994. An evolutionary algorithm that evolves recurrent neural networks. IEEE Transactions on Neural Networks 5 (1), 54–65.
Castillo, P.A., Merelo, J.J., Arenas, M.G., Romero, G., 2007. Comparing evolutionary hybrid systems for design and optimization of multilayer perception structure along training parameters. Information Sciences: An International Journal 177 (14), 2884–2905.
Chaves, P., Chang, F.J., 2008. Intelligent reservoir operation system based on evolving artificial neural networks. Advances in Water Resources 31, 926–936.
Chiang, Y.M., Chang, F.J., Jou, B.J.D., Lin, P.F., 2007. Dynamic ANN for precipitation estimation and forecasting from radar observations. Journal of Hydrology 334, 250–261.
Chiang, Y.M., Chang, L.C., Chang, F.J., 2004. Comparison of static-feedforward and dynamic-feedback neural networks for rainfall–runoff modeling. Journal of Hydrology 290, 297–311.
Chang, F.J., Chang, L.C., Wang, Y.S., 2007. Enforced self-organizing map neural networks for river flood forecasting. Hydrological Processes 21, 741–749.
Chang, Y.T., Chang, L.C., Chang, F.J., 2005. Intelligent control for modeling of real time reservoir operation: part II ANN with operating curves. Hydrological Processes 19, 1431–1444.
Cortez, P., Machado, J., Neves, J., 1996. An evolutionary artificial neural network time series forecasting system. In: Proceedings of the IASTED International Conference on Artificial Intelligence, Expert Systems and Neural Networks, pp. 278–281.
Dawson, C.W., See, L.M., Abrahart, R., Heppenstall, A.J., 2006. Symbiotic adaptive neuro-evolution applied to rainfall–runoff modeling in northern England. Neural Networks 19, 236–247.
Ham, F.M., Kostanic, I., 2001. Principles of Neurocomputing for Science and Engineering. McGraw-Hill.
Holland, J.H., 1975. Adaptation in Natural and Artificial Systems, second ed. Massachusetts Institute of Technology, Cambridge.
Hsu, K.L., Gupta, H.V., Sorooshian, S., 1995. Artificial neural network modeling of the rainfall–runoff process. Water Resources Research 31 (10), 2517–2530.
Karunasinghe, D.S.K., Liong, S.Y., 2006. Chaotic time series prediction with a global model: artificial neural network. Journal of Hydrology 323, 92–105.
Kim, S., Kim, H.S., 2008. Neural networks and genetic algorithm approach for nonlinear evaporation and evapotranspiration modeling. Journal of Hydrology 351, 299–317.
Kwok, T.Y., Yeung, D.Y., 1997. Constructive algorithm for structure learning in feedforward neural networks for regression problems. IEEE Transactions on Neural Networks 3, 630–645.
Leahy, P., Kiely, G., Corcoran, G., 2008. Structural optimisation and input selection of an artificial neural network for river level prediction. Journal of Hydrology 355 (1), 192–201.
Mackey, M.C., Glass, L., 1977. Oscillation and chaos in physiological control systems. Science 197, 287–289.
Moller, M.F., 1993. A scaled conjugate gradient algorithm for fast supervised learning. Neural Networks 6, 525–533.
Moriaty, D.E., Miikkulainen, R., 1998. Forming neural networks through efficient and adaptive coevolution. Evolutionary Computation 5, 373–399.
Sajikumar, N., Thandaveswara, B.S., 1999. A non-linear rainfall–runoff model using an artificial neural network. Journal of Hydrology 216, 32–55.
Sahoo, G.B., Ray, C., 2006. Flow forecasting for a Hawaii stream using rating curves and neural networks. Journal of Hydrology 317, 63–80.
Yao, X., 1993. A review of evolutionary artificial neural networks. International Journal of Intelligent Systems 8, 539–567.
Yao, X., 1999. Evolving artificial neural networks. Proceedings of the IEEE 87 (9), 1423–1447.