International Journal of Neural Systems, Vol. XX, No. X (2021) 1–15

Self-adaptive Particle Swarm Optimization-based Echo State Network for Time Series Prediction

Yu Xue
1. School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, China
2. Engineering Research Center of Digital Forensics, Ministry of Education, Nanjing University of Information Science and Technology, Nanjing, China
E-mail: xueyu@nuist.edu.cn

Qi Zhang
School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, China
E-mail: zhangq@nuist.edu.cn

Ferrante Neri*
COL Laboratory, School of Computer Science, University of Nottingham, Nottingham, UK
E-mail:ferrante.neri@nottingham.ac.uk

Echo state networks (ESNs), belonging to the family of recurrent neural networks (RNNs), are suitable for addressing complex nonlinear tasks due to their rich dynamic characteristics and easy implementation. The reservoir of the ESN is composed of a large number of sparsely connected neurons with randomly generated weight matrices. How to set the structural parameters of the ESN becomes a difficult problem in practical applications. Traditionally, the design of the parameters of the ESN structure is performed manually. The manual adjustment of the ESN parameters is not convenient since it is an extremely challenging and time consuming task. The present paper proposes an ensemble of five particle swarm optimization (PSO) strategies to design the structure of ESN and then reduce the manual intervention in the design process. An adaptive selection mechanism is used for each particle in the evolution to select a strategy from the strategy candidate pool for evolution. In addition, leaky integration neurons are used as reservoir internal neurons, which are added within the adaptive mechanism for optimization. The root mean squared error (RMSE) is adopted as the evaluation criterion. The experimental results on Mackey-Glass time series benchmark dataset show that the proposed method outperforms other traditional evolutionary methods. Furthermore, experimental results on electrocardiogram dataset show that the proposed method on the ensemble of PSO displays an excellent performance on real-world problems.

Keywords: Time Series Prediction, Particle Swarm Optimization, Self-adaptive, Echo State Network, ECG.

1. Introduction

Nature-inspired computing [12; 58] is an established field of computer science where algorithms are designed by following the inspiration of natural phenomena. Some of the most popular sources of inspiration are evolution [6; 32] and biological brains [47]. However, there exist other algorithms inspired by gravitation theory [59], fluid-dynamics [56], music [57], spiral phenomena in nature [55], and cell membranes [76; 83]. These inspired algorithms are used for a number of tasks including for example optimization [8; 7], computational modelling [45; 68], and data science problems [71; 70]. The present paper proposes a nature-inspired approach with reference to time series.

Modeling and forecasting of nonlinear time series play an important role in many engineering problems [28]. For example, time series prediction can be used in intelligent fault detection [36; 48; 51], weather forecast [18; 23; 64], electrical forecast [66; 60], traffic flow prediction [49; 61; 81], energy prediction[21], civil engineering [24], pattern recognition [67; 52] and medical domain [11; 74; 14].

With the rapid growth of artificial intelligence, artificial neural networks (ANNs) [29], due to their capability to deal with nonlinear problems, have gradually become important tools for nonlinear time series prediction [53; 33]. However, traditional ANNs such as recurrent neural networks (RNNs) [17; 75; 46] suffer from the problems of vanishing gradient and exploding gradient and may not perform well due to problems with the slow convergence speed. Among the various types of ANNs, echo state networks (ESN) [26] are proposed. Unlike the traditional RNNs, the ESN only needs to change the weights of the output layer during the training phase. The weights between the reservoir (hidden layer) and the input layer are randomly generated and not altered during the network training process. Thus, the training of an ESN is relatively simple and the calculation cost is small in time and space. Notwithstanding the simplicity in training ESNs can have a performance comparable to that of RNNs. This feature makes ESN an attractive alternative to RNNs for some specific application problems.

Many researchers have successfully applied ESNs to industrial problems and have achieved some promising results in time series prediction. Jaeger [27] applied ESN in wireless communication, and the signal error rate is reduced by two orders of magnitude. Zhang *et al.* [78] proposed a method for data-driven artificial intelligence in fault diagnosis based on the ESN. Ribeiro *et al.* [50] proposed an approach for water flow forecast of hydropower plant which uses extreme learning machines and ESN. Alizamir *et al.* [5] proposed a deep ESN model for soil temperature prediction, and the experimental results show that the algorithm is superior to the traditional three machine learning models mentioned in their literature.

Although ESNs have achieved a considerable success in time series prediction, the fact that the reservoir is randomly initialized causes a certain noise in the ESN performance and may affect it. In order to address this limitation, in the past years, several studies proposed modifications to the ESN structure. Georg *et al.* [15] proposed an ESN with a simple and yet effective reservoir topology, the result shows that a simple structure of ESN can achieve comparable results to classic ESN. Ma *et al.* [38] introduced the idea of deep learning into ESN and added convolution and pooling operations for classification problem, the idea is to combine the advantages of both technologies. Hu *et al.* [22] proposed an ensemble of Bayesian deep ESN models to optimize the parameters of ESN, this approach exhibits a better performance when dealing with more complex time series datasets. Han *et al.* [19] proposed a Laplacian ESN, which overcomes the ill-posed problem due to the small amount of training data and obtains the output weights of low dimension.

Another approach, which became popular with the diffusion of computational intelligence, consists in optimizing the parameters of the ESN. For example, Han *et al.* [41] proposed a modified biogeography-based method to perform the selection of feature subset and the optimization of model parameters. Experimental results on relevant datasets indicate that the algorithm is superior to other traditional evolutionary algorithms, especially in the prediction of multivariable time series. Zhong *et al.* [82] adopted genetic algorithms to optimize the double-reservoir ESN, and the experimental results reveal that the accuracy and stability are excellent in time series datasets compared to other models. Chouikhi *et al.* [13] adopted particle swarm optimization (PSO) [30] to pre-train some fixed weights which are selected in the ESN. Li *et al.* [34] introduced an approach to pre-train a growing ESN with multiple sub-reservoirs by optimizing singular values, based on PSO and singular value decomposition. Both these studies based on PSO displayed an excellent ESN performance with respect to those based on other popular metaheuristics. However, for some specific problems, a single strategy may not be sufficient, so it is necessary to develop a mechanism that can automatically adapt to the needs of the application task.

Inspired by this observation, in this paper we propose an ESN based on self-adaptive particle swarm optimization (SaPSO-ESN) for time series prediction. An adaptive selection mechanism is used for each particle in the evolution to select a strategy from the strategy candidate pool for evolution. In addition, leaky integration neurons are used as reservoir internal neurons, which are added within the adaptive mechanism for optimization. Each individual is assigned an appropriate PSO strategy to make the individual develop towards a lower fitness value. Thus, the convergence speed of the algorithm is accelerated, the optimal solution is found more accurately, and the prediction accuracy is improved. The optimized parameters are then brought into ESN to boot up the network. The root mean squared error (RMSE) is adopted as evaluation criterion. To verify the validity of the proposed algorithm, we performed experiments on Mackey-Glass time series benchmark dataset [40] and electrocardiogram (ECG) time series dataset.

The reminder of this paper is organized as follows. Section 2 elaborates the background knowledge about time series prediction, PSO and ESN. A detailed description of our proposed method SaPSO-ESN is given in Section 3. In Section 4, the proposed algorithm is tested on Mackey-Glass benchmark dataset and ECG dataset. Finally, Section 5 concludes this paper and future research work is presented.

2. Background

Time series is a series of data points indexed in time order. Its units can be seconds, minutes, hours, days, months, years, etc. By analyzing these data to understand past trends and predict future trends. Time series prediction model is mainly based on the existing time series data to make short-term forecasts for the future. This is a kind of complex predictive modeling problem which depends on the past time sequence. Prediction models can be roughly divided into two categories, one is the traditional statistical-based learning method, the other is the more popular machine learning method in recent years. Time series prediction is widely used in stock forecast, weather forecast, agricultural forecast, traffic flow forecast and other fields. Due to the simple and efficient nature, ESN is increasingly used in time series prediction recently.

2.1. *Echo State Network*

ESN is a three-layer recurrent neural network consists of three parts: input layer, reservoir and output layer. Each layer is connected to the other layers by neurons. Each connection has a weight value that forms the weight connection matrix. The neurons in the reservoir are randomly sparsely loop connected, and the structure of ESN is given in Fig. 1.

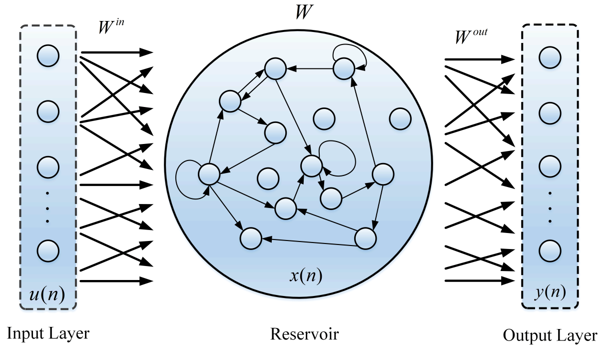

Fig. 1. Structure of a typical ESN

Assuming that ESN with K input neurons, N reservoir neurons, and L output neurons, then at time n, the input $u(n)$, the state of the reservoir $x(n)$ and the output $y(n)$ are shown in the below equations.

$$u(n) = [u_1(n), u_2(n), ..., u_K(n)]^T \tag{1}$$

$$x(n) = [x_1(n), x_2(n), ..., x_N(n)]^T \tag{2}$$

$$y(n) = [y_1(n), y_2(n), ..., y_L(n)]^T \tag{3}$$

Then at time $(n+1)$, the ESN reservoir state can be updated by Eq. (4) and the output equation is represented as Eq. (5) :

$$x(n+1) = f(W^{in}u(n+1) + Wx(n)) \tag{4}$$

$$y(n+1) = f^{out}(W^{out}x(n+1)) \tag{5}$$

where W^{in} and W represent the weight matrix of the input layer and the reservoir, respectively. W^{out} is the weight matrix of the output layer. f and f^{out} are the activation functions of the reservoir and the output layer, respectively. In general, the activation function f of the reservoir is a nonlinear function, such as $tanh$ or $sigmoid$. In this paper, we adopt the $tanh$ as the activation function for the reservoir. f^{out} generally selects the identity function, so the Eq. (5) can be rephrased as Eq. (6).

$$y(n+1) = W^{out}x(n+1) \tag{6}$$

After ESN initialization, only the weights from reservoir to output layer need to be trained by data, while the weights from input layer to reservoir and the weights in the reservoir remain unchanged during the training process. Therefore, the process of solving W^{out} is actually a linear regression process according to Eq. (6). When W^{out} is solved, the predicted value is output according to Eq. (4) and (6).

The performance of the ESN would be better when the reservoir neurons adopt the leaky integrator neurons [37]. In this paper, reservoir neurons with leaky integrator are used, and the state of reservoir can be updated by Eq. (7),

$$x(n+1) = (1-a)x(n) + af(W^{in}u(n+1) + Wx(n)) \tag{7}$$

where a is the leaking rate, it can be regarded as the speed of the reservoir update dynamics discretized in time.

With the purpose of eliminating the impact of the primary state of the reservoir on the network, the previous Q samples were discarded. Q is the number of discarded samples in the training set. Starting from $Q+1$ sample, the corresponding internal state matrix M was collected, and the pseudo-inverse method or ridge regression method was used to solve W^{out}. Ridge regression method is adopted in this paper, Y^{target} is the target output, M is the internal state matrix of ESN, and the solution formulas are shown as follows:

$$Y^{target} = W^{out}M \tag{8}$$

$$W^{out} = Y^{target}M^T(MM^T + \lambda I)^{-1} \tag{9}$$

where λ is the regularization coefficient in the ridge regression, and I is the unit matrix. When W^{out} is trained, the new data will be input into the ESN. After calculation, ESN output the corresponding predicted data.

The core of ESN is the dynamic reservoir, and the performance of the reservoir depends on certain parameters, namely size of reservoir N, spectral radius ρ, sparsity of the reservoir SR, and input scaling IS. For the reservoir with leaky integrator neurons, the leaking rate a also affect ESN performance. Some brief descriptions of these parameters are introduced below:

1) Size of Reservoir (N): It is the size of neurons in the reservoir. A large number of neurons map the input data to the high-dimensional space, and nonlinear fit the expected output.

2) Spectral Radius (ρ): The spectral radius is the maximum value of the eigenvalue absolute value of the reservoir weight matrix W. ESN exhibits echo state property as long as the ρ is at the range of $[0, 1]$.

3) Sparsity of the Reservoir (SR): The reservoir sparsity is the ratio of interconnected neurons to the total number of neurons in the reservoir.

4) Input Scaling (IS): The scaling factor is to scale the input data prior to injection into the reservoir. For W^{in} with different distributions, we should adopt different IS, it usually in the range of $[0, 1]$.

5) Leaking Rate (a): The leaking rate (a) of the leaking neurons in the reservoir can be viewed as the velocity of the reservoir update. The smaller the a, the less dynamic the reservoir becomes, which could improve the short-term memory of the ESN.

2.2. *Particle Swarm Optimization*

PSO is a global stochastic search algorithm [6] based on swarm intelligence proposed by Kennedy and Eberhart [30], which is simulates the migration and clustering behaviors of birds in the foraging process and has been successfully applied in a number of cases [20; 25]. For a population of ps particles, each particle in the search space has a position (x_i) and a velocity (v_i). The velocity of the particle is updated according to its historical optimal position ($Pbest$) and the historical optimal position of the population ($Gbest$). In the iterative process, the velocity and position of the particle are constantly adjusted until the preset conditions are satisfied. The update formula of the d-dimensional of the i-th particle at $t+1$ iteration are as follows:

$$v_{i,d}^{t+1} = v_{i,d}^{t} + c_1 * r_1 * (Pbest_{i,d} - x_{i,d}^{t}) + c_2 * r_2 * (Gbest_d - x_{i,d}^{t}) \tag{10}$$

$$x_{i,d}^{t+1} = x_{i,d}^{t} + v_{i,d}^{t+1} \tag{11}$$

where c_1 and c_2 are acceleration constants, with c_1 is the self-learning factor and c_2 is the group learning factor for each particle. And r_1 and r_2 are two random numbers distributed over $[0, 1]$, d is the dimension of particles, t is the number of iterations and i denotes the current particle. The main framework of PSO is shown in the Fig. 2.

```
Input: Population size ps; number of fitness evaluations nfe; current
number of fitness evaluation cfe;
Output: Position of the approximate global optima Gbest;

Randomly initialize ps particles, including position X_i(0) and velocity V_i(0);
Evaluate the population and set Pbest and Gbest;
while cfe < nfe do
  for i = 1 to ps do
    Update the velocity by Eq. (10);
    Update the position by Eq. (11);
    Calculate its fitness value of particle i : f(X_i);
    if X_i is better than Pbest then
      Pbest = X_i;
      if X_i is better than Gbest then
        Gbest = X_i;
      end if
    end if
    i = i + 1;
  end for
  cfe = cfe + 1;
end while
return Gbest
```

Fig. 2. Pseudo code of the PSO

3. SaPSO-ESN for Parameter Optimization

ESN is characterized by simple training and low computational complexity. However, the setting of reservoir parameters will directly affect the performance of ESN. Manual adjustment of parameters is both time-consuming and does not guarantee that the selected parameters are optimal. Adaptive mechanism has been successfully applied in the field of neural network [72; 73]. Therefore, in this section, we come up with a SaPSO-ESN model for time series prediction, in which an ensemble of PSO is adopted to optimize parameters of ESN. Our goal is to reduce the gap between the target value and the predicted value. Different from traditional PSO, five strategies are adopted to form the strategy candidate pool, which can further enhance the ability of the model to adapt to different problems.

The use of adaptively coordinated multiple search operators/algorithms is a popular strategy in metaheuristic optimization and machine learning. This idea is present in frameworks such as hyper-heuristics [9], memetic algorithms [44], ensemble algorithms [80; 4; 63]. Ensemble algorithms have been successfully implemented in multiple and diverse fields such as traffic speed forecasting [79], rust diagnosis of steel structures [69], and indoor environmental quality [31].

When multiple algorithms are present in a framework, a coordination scheme is necessary. To ensure that the coordination is performed automatically at run time, a popular approach is the employment of a self-adaptation logic. In literature many examples are present in both optimization [2; 3; 43; 42] and machine learning [62; 10; 77].

3.1. *Five PSO Implementation Strategies*

There are many different PSO strategies in related research, and the general structure of these different strategies is similar. They use different formulas and record the experience information to generate new populations. By examining the study of PSO strategies in the existing literature [72 65], we selected the following five strategies for our algorithm, which have been proved to have good performance in the corresponding literature. Five strategies are described as follows.

3.1.1. *PSO with inertia weight*

The original PSO strategies use the position of $Pbest$ and $Gbest$ to update the velocity and position of the particle. In order to enhance the local search capability of the PSO, literature [54] proposed a PSO strategy with inertial weight (PSO-w), where w is the inertial weight, usually taking values between 0 and 1. The updating equation is as follows:

$$v_{i,d}^{t+1} = w*v_{i,d}^{t}+m_1*(Pbest_{i,d}-x_{i,d}^{t})+m_2*(Gbest_d-x_{i,d}^{t}) \quad (12)$$

where m_1 represents $c_1 * r_1$, m_2 represents $c_2 * r_2$.

3.1.2. *PSO with differential idea*

Wang *et al.* [65] proposed an update PSO strategy based on the differential idea (PSO-d). Differential evolution (DE) algorithm is also an efficient global optimization algorithm. PSO-d avoids gradual changes in velocity, but completely updates the velocity based on differential information. The updating equations are described as follows:

$$v_{i,d}^{t+1} = c*(x_{a,d}^{t} - x_{b,d}^{t}) + c*(Pbest_{i,d} - x_{i,d}^{t}) \quad (13)$$

$$c = N(0.5, 0.2) \tag{14}$$

where $x^t_{a,d}$ and $x^t_{b,d}$ are two random particles in the t-th generation population. $N(0.5, 0.2)$ represents a random number that satisfies a Gaussian distribution.

3.1.3. *Local estimation of distribution*

In order to make better performance for PSO, Wang *et al.* [65] introduced a PSO strategy with Gaussian and Cauchy distributions (PSO-l). The equations are expressed as follows:

$$c = \frac{(D-1)N(0,1)}{D} + \frac{C(0,1)}{D} \tag{15}$$

$$z = \sqrt{(Pbest_{i,d} - m^t_{i,d})^2 + (x^t_{i,d} - m^t_{i,d})^2 + (x^t_{k,d} - m^t_{i,d})^2} \tag{16}$$

$$v^{t+1}_{i,d} = (m^t_{i,d} - x^t_{i,d}) + \frac{c}{\sqrt{3}} z \tag{17}$$

where $N(0, 1)$ and $C(0, 1)$ are values generated randomly from the Gaussian and Cauchy distributions, $x^t_{k,d}$ is a random particle choose from the population, and $m^t_{i,d}$ is the average of the best 20% of particles in the population.

3.1.4. *Comprehensive learning PSO*

Liang *et al.* [35] proposed a deformation of the PSO, called comprehensive learning particle swarm optimizer (CLPSO). Unlike the traditional PSO algorithms that only use the own $Pbest$ and $Gbest$ of the particle as directions to guide the flight of the particle, the $Pbest$s of all particles in the proposed algorithm can potentially be the guiding direction of the particles. The equation is expressed as follows:

$$v^{t+1}_{i,d} = w * v^t_{i,d} + c * rand_{i,d} * (Pbest_{f_i(d)} - x^t_{i,d}) \tag{18}$$

where $Pbest_{f_i(d)}$ can be the value of $Pbest$ for any particle.

3.1.5. *An improved CLPSO*

Wang *et al.* [65] improved CLPSO, and an algorithm called PSO-CL-pbest was proposed. The equations are expressed as follows:

$$v^{t+1}_{i,d} = w * v^t_{i,d} + q * (Pbest_{f_i(d)} - x^t_{i,d} + Pbest_{i,d} - x^t_{i,d}) \tag{19}$$

$$q = 0.5 * c * rand_i \tag{20}$$

where $rand_i$ represents an identical random number for update the velocity vector. That is to say, for each dimension of the particle, the same random number $rand_i$ is used to update the velocity.

These strategies have different advantages, and the important parameters setting for the five PSO strategies POS-w, PSO-d, PSO-l, CLPSO and PSO-CL-pbest are shown in the Table 1.

Table 1. Parameters settings for the five PSO strategies

Algorithm	Parameters Setting
PSO-w	$w = 0.9 - \frac{0.5*cfe}{nfe}$, $c_1 = c_2 = 1.49618$
PSO-d	$c = N(0.5, 0.2)$
PSO-l	$c = \frac{(D-1)N(0,1)}{D} + \frac{C(0,1)}{D}$
CLPSO	$w = 0.9 - \frac{0.5*cfe}{nfe}$, $c = 1.49445$
PSO-CL-pbest	$w = 0.9 - \frac{0.5*cfe}{nfe}$, $c = 1.49445$

3.2. ***Strategies Self-adaptive Mechanism***

In traditional PSO algorithms, there is only one evolutionary strategy, meaning that the same strategy is used for the whole population. However, in practical application, different problems have different characteristics, which leads to poor generalization of using only one strategy. In this paper, an adaptive method is used to select strategies in the strategy candidate pool during the population evolution process.

Assuming that the number of strategies in the candidate pool is P, at the beginning of the algorithm, the probability of each strategy being selected in the strategy candidate pool is the same, which is $1/P$, the initialized strategy probability matrix (Pro) is shown in the Eq. (21).

$$Pro = (1/P, 1/P, ..., 1/P)_{1*P} \tag{21}$$

We set a probability update parameter called LP, which means that the probability matrix Pro is updated once after evolution of LP generations. According to the relevant theoretical analysis and experimental results, LP is set to 5 in this paper. Suppose the j-th strategy is selected for the i-th particle (x_i), if the newly generated particle (x^{new}_i) is better than the x_i, the evolution of x_i with the j-th strategy is successful. If the x_i^{new} is worse than the x_i, the particle is failed to evolve into the next generation. In the LP generations, the number of particles which successfully evolved into the next generation through the m-th strategy is denoted as NS_m, and the number of particles which failed to evolve into the next generation is denoted as NF_m. After LP iterations, the probability of the m-th strategy S_m is updated as follows:

$$S_m = \frac{NS_m}{NS_m + NF_m} + \varepsilon \tag{22}$$

To avoid zero probability for the strategy, we set a very small number $\varepsilon = 0.001$, so that we can avoid the situation where $S = 0$. In order to ensure that the sum of the probabilities of all strategies being selected in the strategy pool is 1, we need to normalize all the S obtained, so as to obtain the probability of the selection of the m-th strategy. The final probability of the m-th strategy is shown in the Eq. (23).

$$P_m = \frac{S_m}{\sum_{i=1}^{P} S_i} \tag{23}$$

According to probability matrices Pro after LP generation, the roulette wheel method [16] is used to select the strategy for particles in population.

3.3. *Optimized Parameters*

There are some parameters in ESN that impinge on the network performance. The performance of the ESN would be better when the reservoir neurons adopt the leaky integrator neurons [37]. The parameter of leaky integrator neurons is added to the adaptive mechanism. The method in this paper optimizes five parameters, which are size of reservoir (N), spectral radius (ρ), sparsity of the reservoir (SR), input scaling (IS), and leaking rate (a). These parameters are treated as a particle with five dimensions. The particles of the candidate solution can be expressed as Eq. (24) :

$$x_i = [N, \rho, SR, IS, a] \tag{24}$$

where $i = 1, 2, ..., ps$.

3.4. *The Framework of SaPSO-ESN*

In this paper, the proposed SaPSO-ESN algorithm, which integrates five pso strategies to search the parameters of ESN, is used to practice sequence prediction research, and the pseudo code of SaPSO-ESN is shown in the Fig. 3.

Input: Population size ps; number of fitness evaluations nfe; current number of fitness evaluation ($cfe = 0$); LP; strategy probability matrix Pro;
Output: $Gbest$;

Initial N, ρ, SR, IS, a, and population $(x_1, x_2, ..., x_{ps})$
while $cfe < nfe$ **do**
  for $i = 1$ to ps **do**
    Select the PSO strategy m based on the Pro;
    Generate new velocity (v_i^{new}) and update positions (x_i^{new});
    Create an ESN following x_i^{new};
    Evaluate the fitness value $f(x_i^{new})$ in ESN;
    if x^{new} is better than x_i **then**
      $NS_m = NS_m + 1$;
      if x_i^{new} is better than $Pbest_i$ **then**
        $Pbest_i = x_i^{new}$;
        if x_i^{new} is better than $Gbest$ **then**
          $Gbest = x_i^{new}$;
        end if
      end if
    else
      $NF_m = NF_m + 1$;
    end if
    $i = i + 1$;
    $v_i = v_i^{new}$;
    $x_i = x_i^{new}$;
    $cfe = cfe + 1$;
  end for
  if Iterative LP generation **then**
    Update Pro based on Eq. (22) and (23);
  end if
end while
return $Gbest$

Fig. 3. Pseudo code of the SaPSO-ESN

Firstly, the ESN is encoded into a particle and each particle contains five dimensions, which are $[N, \rho, SR, IS, a]$. Then initializing the population, and the position (x) and velocity (v) of the population in PSO are randomly initialized. The x is in the interval $[x_{min}, x_{max}]$ and v is in the interval $[-v_{max}, v_{max}]$. Before the stop condition is satisfied, a selected strategy is returned to each particle in the population according to the strategy probability matrix Pro and roulette wheel method during the iteration process. According to the selected strategy, population evolve into the next generation. Then create ESNs following the particles in the evolved population, and each individual in population is evaluated and the corresponding fitness value is calculated according to the evaluation criterion. Update the particle of PSO and the records of the success and failure of the evolutionary process (NS_m and NF_m) according to the fitness values. The strategy selection matrix Pro is updated when the LP generations of particles evolve following NS_m and NF_m. Finally, output the optimal individual $Gbest$ and the corresponding results. This output $Gbest$ is the individual that records the optimal ESN searched for. The flowchart of SaPSO-ESN is shown in the Fig. 4.

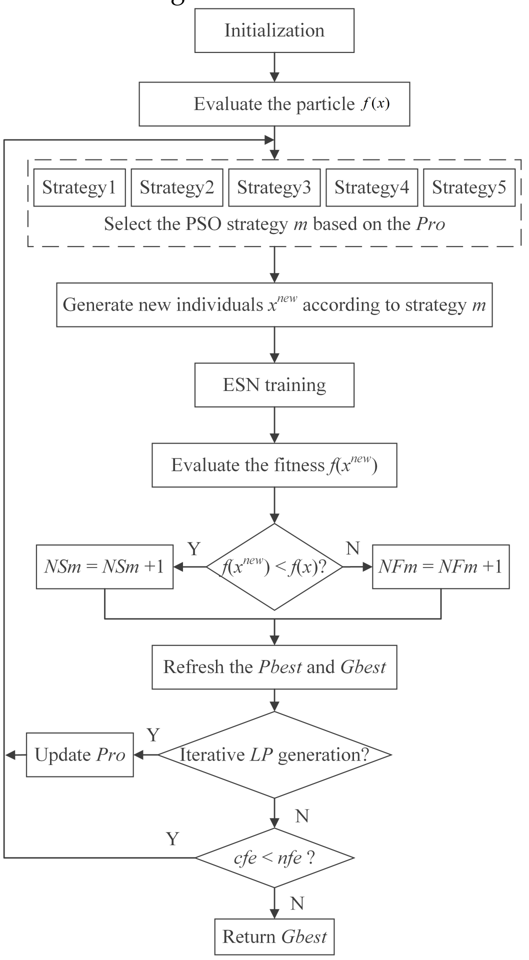

Fig. 4. Flowchart of SaPSO-ESN

4. Experiment

In this section, we evaluate SaPSO-ESN in the benchmark chaotic time series and an ECG datasets. To test the effectiveness of the proposed method in this paper, we select some relevant algorithms for comparison, such as canonical ESN, PSO and DE. In order to demonstrate the effectiveness of the proposed self-adaptive mechanism, we add the comparison results between SaPSO-ESN and the algorithm where five PSO strategies are selected randomly (named RPSO-ESN) in the evolution process. Moreover, SaPSO-ESN is also compared with MTLBO [39] proposed in recent years.

For the traditional ESN, we initialize 30 ESN networks randomly to take the average value as a comparison. To guarantee a fair comparison, we set the number of function evaluations (NFE) as the stopping criterion for every algorithm. In this paper, we set the population size to 100, the number of iterations is 100, which means the NFE is 10000. The reservoir size N is set to $[20, 100]$, spectral radius ρ is set to $[0.1, 1]$, sparsity of the reservoir SR is set to [0.01,0.5], input scaling IS is set to $[0.001, 1]$, and leaking rate a is set to $[0.1, 1]$. In order to eliminate randomness, our experiment is repeated 30 times to take the average.

4.1. *Performance Evaluation Index*

In this paper, we use RMSE to evaluate the performance of the algorithm. The related formula is expressed in Eq. (25).

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y^{target}(t) - y(t))^2} \tag{25}$$

where $y^{target}(t)$ and $y(t)$ represent target values and network output values at time t, respectively. n represents the size of sample points in the test set.

4.2. *Mackey-Glass Time Series*

Mackey-Glass chaotic system (MGS) [40] is a kind of typical chaotic system, and the model is described by the following equation:

$$\frac{\mathrm{d}y(t)}{\mathrm{d}(t)} = \frac{ay(t-\tau)}{1+y^c(t-\tau)} - by(t) \tag{26}$$

where the value of a, b, c are set to 0.2, 0.1, and 10 in many cases. MGS exhibits some sort of periodicity ($\tau < 16.8$) and chaos ($\tau > 16.8$) depends on the value of τ. The most used τ in the literature are $\tau = 17$ and $\tau = 30$.

In the experimental results of the MGS, we set $\tau = 17$ and $\tau = 30$, in other words, the MGS exhibits chaotic characteristics at these situations. We use the Eq. (26) to activate 1000 sample points, of which 500 samples are used as the training set of ESN and 500 samples make up the test set. A graph of MGS with $\tau = 17$ is given in Fig. 5, and $\tau = 30$ is given in Fig. 6. To offset the effect of the initial state reservoir on the results, we discarded the first 50 input data to clean the reservoir.

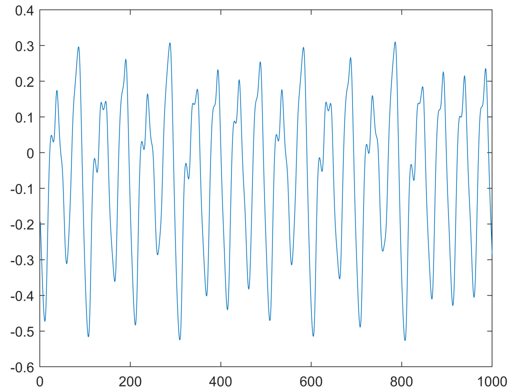

Fig. 5. Mackey-Glass time series ($\tau = 17$)

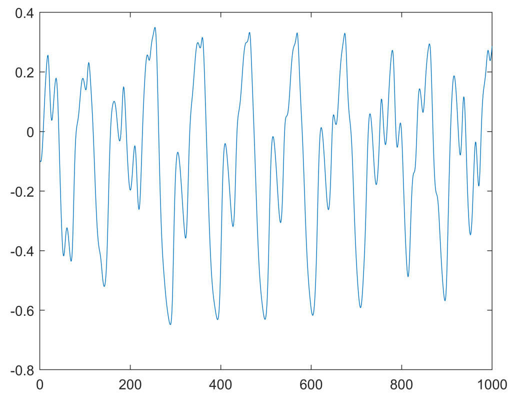

Fig. 6. Mackey-Glass time series ($\tau = 30$)

Fig. 7 and Fig. 8 show the gap between the target signal and the network outputs signal in MGS time series. As the data is small, it can be seen from the figure that the predicted value is close to the target value. Table 2 gives the best parameters of ESN selected by SaPSO-ESN for the MGS time series. Table 3 presents the prediction results of MGS time series in different algorithms, with the evaluation criterion RMSE. It can be seen from the table 3 that the RMSE of the three methods is smaller than that of the traditional ESN, and the RMSE of our proposed method is the smallest. The RMSE of SaPSO-ESN is smaller than that of RPSO-ESN in the MGS, proving that our adaptive mechanism is effective. DE-ESN performs better than PSO-ESN, and MTLBO-ESN performs almost the same as DE-ESN. The RMSE of MGS-17 is less than MGS-30, which indicates that the prediction task is more difficult for MGS-30.

Table 2. Parameters of ESN selected by SaPSO-ESN on MGS

Parameters	$\tau = 17$	$\tau = 30$
Size of reservoir	97	84
Spectral radius	0.9912	0.8132
Sparsity of reservoir	0.2828	0.3602
Input scaling	0.6343	0.3213
Leaking rate	0.9913	0.9999

Table 3. The prediction results on MGS (RMSE)

Method	$\tau = 17$	$\tau = 30$
Traditional ESN	6.72e-04	2.05e-03
PSO-ESN	8.26e-05	2.27e-04
DE-ESN	6.84e-05	2.11e-04
MTLBO-ESN	6.88e-05	2.15e-04
RPSO-ESN	6.18e-05	1.99e-04
SaPSO-ESN	5.94e-05	1.92e-04

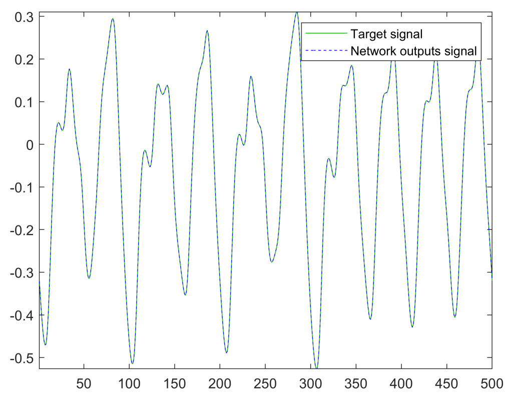

Fig. 7. The target signals VS SaPSO-ESN generated signals (Mackey-Glass time series with $\tau = 17$)

Fig. 9 and 10 are the fitness curves of different models. Compared with the single strategy PSO algorithm PSO-ESN, in Fig. 9, SaPSO-ESN has a slower convergence speed but better prediction result. In Fig. 10, the results are slightly different. As SaPSO-ESN not only has the same convergence speed as PSO-ESN, but also has the least prediction accuracy. MTLBO-ESN converges fastest as is shown in Fig. 10. In other words, for more complex tasks, SaPSO-ESN may have better performance.

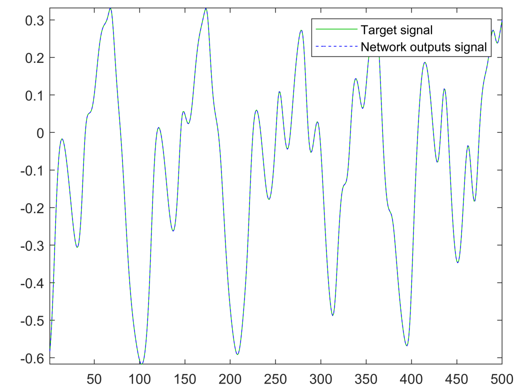

Fig. 8. The target signals VS SaPSO-ESN generated signals (Mackey-Glass time series with $\tau = 30$)

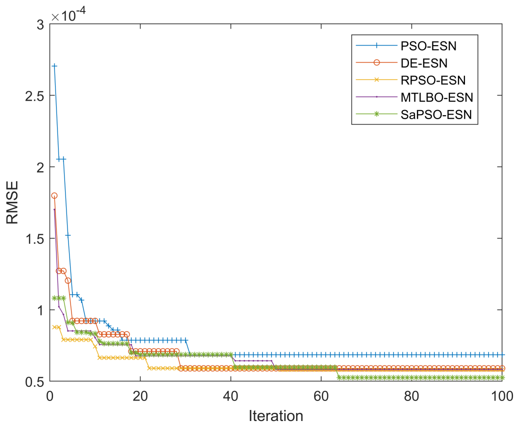

Fig. 9. Fitness curves of different models (Mackey-Glass time series with $\tau = 17$)

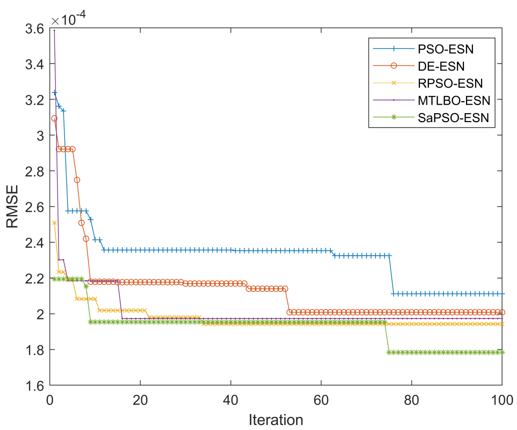

Fig. 10. Fitness curves of different models (Mackey-Glass time series with $\tau = 30$)

4.3. *ECG Datasets*

The human body is a mixed whole containing a large number of linear and nonlinear systems, and the heart is one of the most complex nonlinear systems. Many studies have shown that the physiology of the heart is neither periodic nor completely random, but chaotic. ECG signals are composed of a series of characteristic waves, which contain a wealth of pathological knowledge. ECG signals can be used to detect arrhythmias, myocardial infarction, abnormal heart rate, electrolyte disturbance, heart failure and other conditions. If we can predict the movement trend of ECG, we can predict the disease in advance and achieve early intervention treatment, and avoid many tragedies. With the proliferation of wearable devices, ECG signals have become easier to gather, so this gives us a lot of space for future research.

The ECG datasets used in this paper is collected by a hospital sleep monitoring center, with a total of 10 channels of data, and ECG signals are also collected. The sampling rate is 512 Hz, which means that there are 512 sample points in one second. The ECG signal is shown in Fig. 11 with 1000 sample points, 500 samples for training, 500 samples for testing, and 50 samples for washing the reservoir. In order to eliminate the randomness, we repeat the test 30 times, and then take the average value as the test result. Table 4 gives the best parameters of ESN selected by SaPSO-ESN for the ECG datasets. Table 5 gives the one-step prediction results of different models on ECG datasets. So the model proposed in this paper is better than other models of comparison in RMSE.

Table 4. The best parameters of ESN selected by SaPSO-ESN on ECG datasets

Parameters	Value
Size of reservoir	76
Spectral radius	0.6129
Sparsity of reservoir	0.2509
Input scaling	0.8290
Leaking rate	0.8669

Table 5. The prediction results on ECG datasets

Method	RMSE
Traditional ESN	5.85e-03
PSO-ESN	3.30e-03
DE-ESN	3.20e-03
MTLBO-ESN	3.29e-03
RPSO-ESN	2.91e-03
SaPSO-ESN	2.84e-03

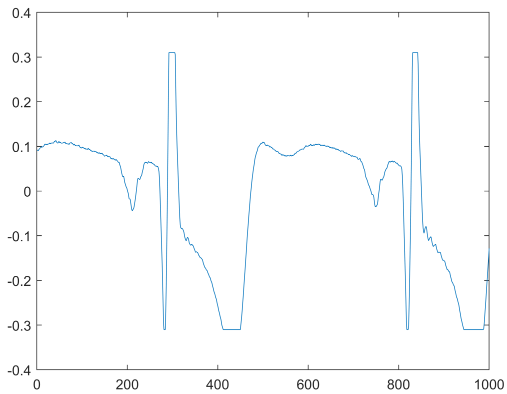

Fig. 11. 1000 samples of ECG datasets

Fig. 12 shows the predictive curve of SaPSO-ESN to EEG datasets, and Fig. 13 shows the error curves of different algorithms. In order to make the contrast more obvious, we enlarge the key parts in Fig. 12. As can be seen in the figure, the prediction effect will get worse when it is at the boundary point of the curve. As is shown in Fig. 12, SaPSO-ESN in this paper can fit ECG data well. Moreover, from Fig. 13, all the three algorithms converged before 50 generations. The convergence speed of SaPSO-ESN is the fastest, about 17 generations, while that of DE-ESN is the slowest, about 30 generations, and that of PSO-ESN is between the two method, about 25 generations. Although the convergence speed of PSO-ESN is faster than that of DE-ESN, the prediction performance is not as good as that of DE-ESN. It also shows that the SaPSO-ESN proposed by us has better convergence speed and prediction performance in complex real applications.

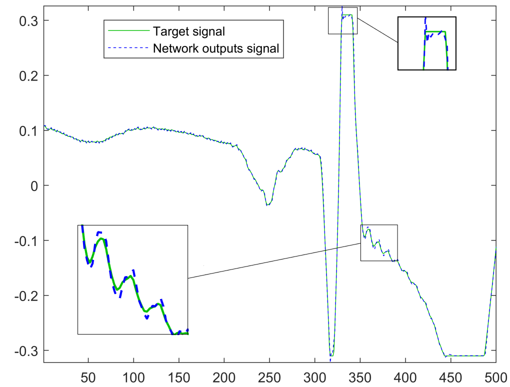

Fig. 12. The prediction curve of SaPSO-ESN on ECG datasets

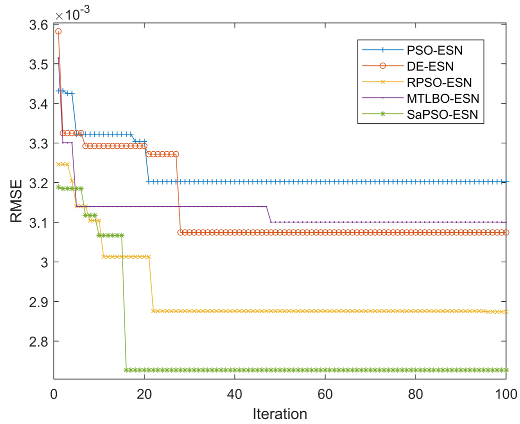

Fig. 13. Fitness curves of different models on ECG datasets

5. Conclusion

In this paper, we use an adaptive PSO-based algorithm to dynamically adjust the parameters of ESN for different time series prediction application, so as to improve the prediction accuracy and enhance the generalization. There are two main improvements in our algorithm. One is that we adopt the leaky integrator neurons with adaptive parameters in the ESN, and the leaking rate changes according to the training process. The other is the adaptive PSO strategy. Experimental results on Mackey-Glass time series and ECG signals show that the proposed algorithm has considerable improvement and has a fast convergence rate. For future work, we intend to mix different evolutionary computing methods on the basis of adaptive frameworks, not just PSO. This work can also be used in the prediction of electroencephalogram and electromyography, which play an key role in the prevention of diseases and reducing the labor intensity of medical workers.

Acknowledgment

This work was partially supported by the National Natural Science Foundation of China (61876089, 61876185, 61902281), the Opening Project of Jiangsu Key Laboratory of Data Science and Smart Software (No.2019DS301), the Natural Science Foundation of Jiangsu Province (BK20141005), the Natural Science Foundation of the Jiangsu Higher Education Institutions of China (14KJB520025), and the Priority Academic Program Development of Jiangsu Higher Education Institutions.

References

1. *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2013, Cancun, Mexico, June 20-23, 2013* (IEEE, 2013).
2. H. Adeli and S. Hung, An adaptive conjugate gradient learning algorithm for efficient training of neural networks, *Applied Mathematics and Computation* **62** (April 1994) 81–102.
3. H. Adeli and A. Saleh, Integrated structural/control optimization of large adaptive/smart structures, *International Journal of Solids and Structures* **35** (October 1998) 3815–3830.
4. K. M. R. Alam, N. Siddique and H. Adeli, A dynamic ensemble learning algorithm for neural networks, *Neural Computing and Applications* **32** (June 2020) 8675–8690.
5. M. Alizamir, S. Kim, M. Zounemat-Kermani, S. Heddam, A. H. Shahrabadi and B. Gharabaghi, Modelling daily soil temperature by hydro-meteorological data at different depths using a novel data-intelligence model: deep echo state network model, *Artificial Intelligence Review* (2020) 1–28.
6. C. Blum, R. Chiong, M. Clerc, K. A. D. Jong, Z. Michalewicz, F. Neri and T. Weise, Evolutionary optimization, *Variants of Evolutionary Algorithms for Real-World Applications*, eds. R. Chiong, T. Weise and Z. Michalewicz (Springer, 2012), pp. 1–29.
7. F. Caraffini, G. Iacca, F. Neri, L. Picinali and E. Mininno, A CMA-ES super-fit scheme for the resampled inheritance search, *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2013, Cancun, Mexico, June 20-23, 2013*, (IEEE, 2013), pp. 1123–1130.
8. F. Caraffini, F. Neri, J. Cheng, G. Zhang, L. Picinali, G. Iacca and E. Mininno, Super-fit multicriteria adaptive differential evolution, *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2013, Cancun, Mexico, June 20-23, 2013*, (IEEE, 2013), pp. 1678–1685.
9. F. Caraffini, F. Neri and M. G. Epitropakis, Hyperspam: A study on hyper-heuristic coordination strategies in the continuous domain, *Inf. Sci.* **477** (2019) 186–202.
10. M. Chiachío, J. Chiachío, D. Prescott and J. Andrews, Plausible Petri nets as self-adaptive expert systems: A tool for infrastructure asset monitoring, *Computer-Aided Civil and Infrastructure Engineering* (December 2018) p. mice.12427.
11. A. M. Chiarelli, P. Croce, G. Assenza, A. Merla, G. Granata, N. M. Giannantoni, V. Pizzella, F. Tecchio and F. Zappasodi, Electroencephalography-derived prognosis of functional recovery in acute stroke through machine learning approaches, *International Journal of Neural Systems* (2020) 2050067–2050067.
12. R. Chiong, F. Neri and R. I. McKay, Nature that breeds solutions, *Int. J. Signs Semiot. Syst.* **2**(2) (2012) 23–44.
13. N. Chouikhi, B. Ammar, N. Rokbani and A. M. Alimi, Pso-based analysis of echo state network parameters for time series forecasting, *Applied Soft Computing* **55** (2017) 211–225.
14. B. Direito, C. A. Teixeira, F. Sales, M. Castelo-Branco and A. Dourado, A realistic seizure prediction study based on multiclass svm, *International Journal of Neural Systems* **27**(03) (2017) p. 1750006.
15. G. Fette and J. Eggert, Short term memory and pattern matching with simple echo state networks, *International Conference on Artificial Neural Networks*, 2005, pp. 13–18.
16. D. B. Fogel, An introduction to simulated evolutionary optimization, *IEEE transactions on neural networks* **5**(1) (1994) 3–14.
17. A. Graves, Supervised sequence labelling with recurrent neural networks, *Studies in Computational Intelligence* **385** (2012).
18. A. Haidar and B. Verma, A novel approach for optimizing climate features and network parameters in rainfall forecasting, *Soft Computing* **22**(24) (2018) 8119–8130.
19. M. Han and M. Xu, Laplacian echo state network for multivariate time series prediction, *IEEE Transactions on Neural Networks and Learning Systems* **29**(1) (2018) 238–244.
20. S. I. Hossain, M. A. H. Akhand, M. I. R. Shuvo, N. H. Siddique and H. Adeli, Optimization of university course scheduling problem using particle swarm optimization with selective search, *Expert Systems with Applications* **127** (2019) 9–24.
21. H. Hu, L. Wang and S.-X. Lv, Forecasting energy consumption and wind power generation using deep echo state network, *Renewable Energy* **154** (2020) 598–613.
22. R. Hu, Z. R. Tang, X. Song, J. Luo and S. Chang, Ensemble echo network with deep architecture for time-series modeling, *Neural Computing and Applications*

33(10) (2021) 4997–5010.

23. B. Huang, G. Qin, R. Zhao, Q. Wu and A. Shahriari, Recursive bayesian echo state network with an adaptive inflation factor for temperature prediction, *Neural Computing and Applications* **29**(12) (2018) 1535–1543.
24. C. S. Huang, Q. T. Le, W. C. Su and C. H. Chen, Wavelet-based approach of time series model for modal identification of a bridge with incomplete input, *Computer-Aided Civil and Infrastructure Engineering* **35** (September 2020) 947–964.
25. G. Iacca, F. Caraffini and F. Neri, Multi-strategy coevolving aging particle optimization, *International Journal of Neural Systems* **24**(1) (2014).
26. H. Jaeger, Adaptive nonlinear system identification with echo state networks, *Advances in Neural Information Processing Systems* **15** (2002) 609–616.
27. H. Jaeger and H. Haas, Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication, *Science* **304**(5667) (2004) 78–80.
28. X. Jiang and H. Adeli, Fuzzy clustering approach for accurate embedding dimension identification in chaotic time series, *Integrated Computer-Aided Engineering* **10**(3) (2003) 287–302.
29. Y. Kara, M. Boyacioglu and O. Baykan, Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the istanbul stock exchange, *Expert Systems with Applications* **38**(5) (2011) 5311–5319.
30. J. Kennedy and R. Eberhart, Particle swarm optimization, *Proceedings of International Conference on Neural Networks*, **4** 1995, pp. 1942–1948.
31. J. Kim, H. Kim and T. Hong, Automated classification of indoor environmental quality control using stacked ensembles based on electroencephalograms, *Computer-Aided Civil and Infrastructure Engineering* **35** (May 2020) 448–464.
32. M. Kociecki and H. Adeli, Shape optimization of free-form steel space-frame roof structures with complex geometries using evolutionary computing, *Engineering Applications of Artificial Intelligence* **38** (February 2015) 168–182.
33. P. Lara-Benítez, M. Carranza-García and J. C. Riquelme, An experimental review on deep learning architectures for time series forecasting, *International Journal of Neural Systems* **31**(3) (2021) 2130001:1–2130001:28.
34. Y. Li and F. Li, Pso-based growing echo state network, *Applied Soft Computing* **85** (2019) p. 105774.
35. J. J. Liang, A. K. Qin, P. N. Suganthan and S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, *IEEE Transactions on Evolutionary Computation* **10**(3) (2006) 281–295.
36. J. Long, S. Zhang and C. Li, Evolving deep echo state networks for intelligent fault diagnosis, *IEEE Transactions on Industrial Informatics* **16**(7) (2020) 4928–4937.
37. M. Lukovsevivcius, A practical guide to applying echo state networks, *Neural Networks: Tricks of the Trade: Second Edition* (2012) 659–686.
38. Q. Ma, E. Chen, Z. Lin, J. Yan, Z. Yu and W. Y. W. Ng, Convolutional multitimescale echo state network, *IEEE transactions on cybernetics* (2019) 1–13.
39. Y. Ma, X. Zhang, J. Song and L. Chen, A modified teaching–learning-based optimization algorithm for solving optimization problem, *Knowledge-Based Systems* **212** (2021) p. 106599.
40. M. C. Mackey and L. Glass, Oscillation and chaos in physiological control systems, *Science* **197**(4300) (1977) 287–289.
41. X. Na, M. Han, W. Ren and K. Zhong, Modified bbo-based multivariate time-series prediction system with feature subset selection and model parameter optimization, *IEEE Transactions on Cybernetics* (2020) 1–11.
42. F. Neri, Adaptive covariance pattern search, *Applications of Evolutionary Computation - 24th International Conference, EvoApplications 2021, Held as Part of EvoStar 2021, Virtual Event, April 7-9, 2021, Proceedings*, eds. P. A. Castillo and J. L. J. Laredo *Lecture Notes in Computer Science* **12694**, (Springer, 2021), pp. 178–193.
43. F. Neri and S. Rostami, Generalised pattern search based on covariance matrix diagonalisation, *SN Comput. Sci.* **2**(3) (2021) p. 171.
44. F. Neri, V. Tirronen, T. Kärkkäinen and T. Rossi, Fitness diversity based adaptation in multimeme algorithms: A comparative study, *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2007, 25-28 September 2007, Singapore*, (IEEE, 2007), pp. 2374–2381.
45. L. Pan, G. Paun, G. Zhang and F. Neri, Spiking neural *P* systems with communication on request, *Int. J. Neural Syst.* **27**(8) (2017) 1750042:1–1750042:13.
46. A. Panakkat and H. Adeli, Recurrent Neural Network for Approximate Earthquake Time and Location Prediction Using Multiple Seismicity Indicators, *Computer-Aided Civil and Infrastructure Engineering* **24** (May 2009) 280–292.
47. H. S. Park and H. Adeli, Distributed Neural Dynamics Algorithms for Optimization of Large Steel Structures, *Journal of Structural Engineering* **123** (July 1997) 880–888.
48. Z. Pu, C. Li, S. Zhang and Y. Bai, Fault diagnosis for wind turbine gearboxes by using deep enhanced fusion network, *IEEE Transactions on Instrumentation and Measurement* **70** (2020) 1–11.
49. L. Qin, W. Li and S. Li, Effective passenger flow forecasting using stl and esn based on two improvement strategies, *Neurocomputing* **356** (2019) 244–256.
50. V. H. A. Ribeiro, G. Reynoso-Meza and H. V. Siqueira, Multi-objective ensembles of echo state networks and extreme learning machines for streamflow series forecasting, *Engineering Applications of Artificial Intelligence* **95** (2020) p. 103910.
51. M. Rigamonti, P. Baraldi, E. Zio, I. Roychoudhury,

K. Goebel and S. Poll, Ensemble of optimized echo state networks for remaining useful life prediction, *Neurocomputing* **281** (2018) 121–138.

52. J. L. Rosselló, M. L. Alomar, A. Morro, A. Oliver and V. Canals, High-density liquid-state machine circuitry for time-series forecasting, *International Journal of Neural Systems* **26**(05) (2016) p. 1550036.
53. F. Saâdaoui and O. B. Messaoud, Multiscaled neural autoregressive distributed lag: A new empirical mode decomposition model for nonlinear time series forecasting, *International Journal of Neural Systems* **30**(8) (2020) 2050039:1–2050039:15.
54. Y. Shi, A modified particle swarm optimizer, *1998 IEEE International Conference on Evolutionary Computation Proceedings*, 1998.
55. N. Siddique and H. Adeli, Spiral Dynamics Algorithm, *International Journal on Artificial Intelligence Tools* **23** (December 2014) p. 1430001.
56. N. Siddique and H. Adeli, Water Drop Algorithms, *International Journal on Artificial Intelligence Tools* **23** (December 2014) p. 1430002.
57. N. Siddique and H. Adeli, Harmony Search Algorithm and its Variants, *International Journal of Pattern Recognition and Artificial Intelligence* **29** (December 2015) p. 1539001.
58. N. Siddique and H. Adeli, Nature Inspired Computing: An Overview and Some Future Directions, *Cognitive Computation* **7** (December 2015) 706–714.
59. N. Siddique and H. Adeli, Gravitational Search Algorithm and Its Variants, *International Journal of Pattern Recognition and Artificial Intelligence* **30** (September 2016) p. 1639001.
60. H. Siqueira, L. Boccato, R. Attux and C. Lyra, Unorganized machines for seasonal streamflow series forecasting, *International Journal of Neural Systems* **24**(03) (2014) p. 1430009.
61. Z. Song, K. Wu and J. Shao, Destination prediction using deep echo state network, *Neurocomputing* **406** (2020) 343–353.
62. Y. Su, Y. Wu, G. Qiao and S. Shen, Self-adaptive form generation method for reciprocal grid structures, *Computer-Aided Civil and Infrastructure Engineering* **34** (May 2019) 444–454.
63. J. Sun, H. Li and H. Adeli, Concept Drift-Oriented Adaptive and Dynamic Support Vector Machine Ensemble With Time Window in Corporate Financial Risk Prediction, *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **43** (July 2013) 801–813.
64. Z. Tang, G. Zhao and T. Ouyang, Two-phase deep learning model for short-term wind direction forecasting, *Renewable Energy* **173** (2021) 1005–1016.
65. Y. Wang, B. Li, T. Weise, J. Wang, B. Yuan and Q. Tian, Self-adaptive learning based particle swarm optimization, *Information Sciences* **181**(20) (2011) 4515–4538.
66. L. Wei and L. Haitian, Electrical load forecasting using echo state network and optimizing by pso algorithm, *2017 10th International Conference on Intelligent Computation Technology and Automation*, 2017, pp. 394–397.
67. A. J. Wootton, S. L. Taylor, C. R. Day and P. W. Haycock, Optimizing echo state networks for static pattern recognition, *Cognitive Computation* **9**(3) (2017) 1–9.
68. T. Wu, F. Bîlbîe, A. Paun, L. Pan and F. Neri, Simplified and yet turing universal spiking neural P systems with communication on request, *Int. J. Neural Syst.* **28**(8) (2018) 1850013:1–1850013:19.
69. J. Xu, C. Gui and Q. Han, Recognition of rust grade and rust ratio of steel structures based on ensembled convolutional neural network, *Computer-Aided Civil and Infrastructure Engineering* **35** (October 2020) 1160–1174.
70. Y. Xue, P. Jiang, F. Neri and J. Liang, A multiobjective evolutionary approach based on graph-in-graph for neural architecture search of convolutional neural networks, *International Journal of Neural Systems* **31** (2021).
71. Y. Xue, Y. Tang, X. Xu, J. Liang and F. Neri, Multi-objective feature selection with missing data in classification, *IEEE Transactions on Emerging Topics in Computational Intelligence* (2021) 1–10, to appear.
72. Y. Xue, B. Xue and M. Zhang, Self-adaptive particle swarm optimization for large-scale feature selection in classification, *ACM Transactions on Knowledge Discovery from Data* **13**(5) (2019) 1–27.
73. Y. Xue, H. Zhu, J. Liang and A. Słowik, Adaptive crossover operator based multi-objective binary genetic algorithm for feature selection in classification, *Knowledge-Based Systems* (2021) p. 107218.
74. S. Yuan, W. Zhou and L. Chen, Epileptic seizure prediction using diffusion distance and bayesian linear discriminate analysis on intracranial eeg, *International Journal of Neural Systems* **28**(01) (2018) p. 1750043.
75. A. Zhang, K. C. P. Wang, Y. Fei, Y. Liu, C. Chen, G. Yang, J. Q. Li, E. Yang and S. Qiu, Automated Pixel-Level Pavement Crack Detection on 3D Asphalt Surfaces with a Recurrent Neural Network: Automated pixel-level pavement crack detection on 3D asphalt surfaces using CrackNet-R, *Computer-Aided Civil and Infrastructure Engineering* **34** (March 2019) 213–229.
76. G. Zhang, H. Rong, P. Paul, Y. He, F. Neri and M. J. Pérez-Jiménez, A complete arithmetic calculator constructed from spiking neural P systems and its application to information fusion, *Int. J. Neural Syst.* **31**(1) (2021) 2050055:1–2050055:17.
77. J. Zhang, M. Xiao, L. Gao and S. Chu, Probability and interval hybrid reliability analysis based on adaptive local approximation of projection outlines using support vector machine, *Computer-Aided Civil and Infrastructure Engineering* **34** (November 2019) 991–1009.
78. S. Zhang, X. Duan, C. Li and M. Liang, Pre-classified reservoir computing for the fault diagnosis of 3d printers, *Mechanical Systems and Signal Processing* **146** (2021) p. 106961.

79. S. Zhang, L. Zhou, X. M. Chen, L. Zhang, L. Li and M. Li, Network-wide traffic speed forecasting: 3D convolutional neural network with ensemble empirical mode decomposition, *Computer-Aided Civil and Infrastructure Engineering* **35** (October 2020) 1132–1147.
80. X. Zhang, S. T. Waller and P. Jiang, An ensemble machine learning-based modeling framework for analysis of traffic crash frequency, *Computer-Aided Civil and Infrastructure Engineering* **35** (March 2020) 258–276.
81. Y. Zhang, T. Cheng and Y. Ren, A graph deep learning method for short-term traffic forecasting on large road networks, *Computer-Aided Civil and Infrastructure Engineering* **34**(10) (2019) 877–896.
82. S. Zhong, X. Xie, L. Lin and F. Wang, Genetic algorithm optimized double-reservoir echo state network for multi-regime time series prediction, *Neurocomputing* **238** (2017) 191–204.
83. M. Zhu, Q. Yang, J. Dong, G. Zhang, X. Gou, H. Rong, P. Paul and F. Neri, An adaptive optimization spiking neural P system for binary problems, *Int. J. Neural Syst.* **31**(1) (2021) 2050054:1–2050054:17.