Neural Computing and Applications (2023) 35:21697–21707
https://doi.org/10.1007/s00521-023-08942-x

ORIGINAL ARTICLE

Neuro-evolutionary for time series forecasting and its application in hourly energy consumption prediction

Nguyen Ngoc Son[1] · Nguyen Van Cuong[2]

Received: 17 October 2022 / Accepted: 1 August 2023 / Published online: 19 August 2023

Abstract

This paper proposed an ensemble methodology comprising neural networks, modified differential evolution algorithm and nonlinear autoregressive network with exogenous inputs (NARX) (called neuro-evolutionary NARX or NE-NARX model) for time series forecasting. In NE-NARX, the structure is designed by connecting the neural model and NARX model, and the weight value connection is optimized by a modified differential evolution algorithm. The effectiveness of the proposed NE-NARX model is tested on two well-known benchmark datasets, including the Canadian lynx and the Wolf sunspot. The proposed model is compared to other models, including the classical backpropagation algorithm, particle swarm optimization, differential evolution (DE) and DE variants. Additionally, an ARIMA model is employed as the benchmark for evaluating the capacity of the proposed model. And then, NE-NARX model is used for hourly energy consumption prediction through comparison with other machine learning models including gated recurrent units, convolutional neural networks (CNN), long short-term memory (LSTM), a hybrid CNN-LSTM and sequence-to-sequence learning. Results show convincingly the superiority of the proposed NE-NARX model over other models.

Keywords Neuro-evolutionary · ARIMA · Differential evolution · GRU · LSTM · CNN · Sequence-to-sequence learning

Abbreviations

GRU	Gated recurrent units
CNN	Convolutional neural network
LSTM	Long short-term memory
Seq2Seq	Sequence-to-sequence
NARX	Nonlinear autoregressive network with exogenous inputs
NE	Neuro-evolutionary
ARIMA	Autoregressive integrated moving average
DE	Differential evolution
PSO	Particle swarm optimization
MSE	Mean square error
MAPE	Mean absolute percent error

1 Introduction

Various traditional techniques have been used widely in time series modeling and forecasting. For example, Box–Jenkins methods (i.e., autoregressive integrated moving average—ARIMA, autoregressive moving average with exogenous inputs—ARMAX, etc.) were widely applied in the predictive energy management of hybrid electric vehicles [1], day-ahead wind speed forecasting [2], mid-long-term electric energy consumption [3] and so on. However, these traditional time series modeling methods are the preassumed linear model. The approximation of linear models to a complex problem is not always satisfactory, provides reasonable accuracy and suffers from the assumptions of stationarity and linearity.

To deal with accurate forecasting of nonlinear data, machine learning (ML) is introduced as an efficient tool for modeling and forecasting as black-box models, such as gated recurrent units (GRU), long short-term memory (LSTM), convolutional neural networks (CNN) and sequence-to-sequence learning. Among those machine learning (ML) models, an ensemble methodology comprising evolutionary computations (EC) and artificial neural networks (ANN) have the most interest. For the ANN

✉ Nguyen Ngoc Son
nguyenngocson@iuh.edu.vn

Nguyen Van Cuong
nguyenvancuong@iuh.edu.vn

[1] Faculty of Electronics Technology, Industrial University of Ho Chi Minh City, Ho Chi Minh City, Viet Nam

[2] IUQ, Industrial University of Ho Chi Minh City, Ho Chi Minh City, Viet Nam

studies, a feedforward neural network (FNN) was proposed for time series forecasting such as prediction and optimization of heating and cooling loads in a residential building [4], prediction of energy consumption for high buildings [5] and forecast of electricity prices [6]. A convolution neural network (CNN) was introduced for day-ahead spatiotemporal wind speed forecasting [7], robust forecasting of river flow [8] and so on. A recurrent neural network (RNN) is proposed for forecasting across time series databases [9]. Paper [10] presented a current status and future directions analysis of RNN in time series forecasting. A combination of extreme learning machine and Elman neural network was introduced for short-term wind speed prediction [11]. However, the weights of ANN are optimized by the gradient descent method, which has the problem of slow convergence and local optimum.

Recently, evolutionary computations (EC) have been widely implemented in optimizing neural networks because the advantages of EC are global optimization methods and scale well to higher-dimensional problems. For example, Zhang et al. [12] introduced an ensemble model comprising a support vector regression, a chaotic krill herd algorithm and empirical mode decomposition in forecasting. Feng et al. [13] proposed a hybrid neural network and cooperation search algorithm (CSA) for river flow time series forecasting. Talaat et al. [14] presented a combination of a multilayered neural network and the grasshopper optimization algorithm (GOA) for load forecasting. Cai et al. [15] proposed a hybrid neural network and particle swarm optimization (PSO) for short-term traffic flow forecasting. Kılıc et al. [16] introduced a hybrid neural network and evolutionary algorithm for global radiation forecasting.

Among evolutionary computations (EC), differential evolution (DE) [17] and variants were very interesting and applied in many fields such as parameter extraction of photovoltaic models [18, 19], optimizing wind farm layout [20], electricity consumption forecasting [21, 22], an ensemble with neural networks [23, 24], an ensemble with fuzzy logic model [25, 26] and hybrid with other algorithms [27, 28]. To improve the classical DE algorithm, DE variants are focusing on the modified mutation phase, adaptive control parameters (i.e., mutant parameter F, crossover parameter CR, population size NP) and hybrid with other algorithms. For example, paper [18] proposed a self-adaptive ensemble-based differential evolution mutation strategy. Paper [19] proposed a multi-population parallel co-evolutionary mutation strategy of the DE algorithm. Paper [27] introduced a hybrid method based on the cuckoo search (CS) and differential evolution (DE) algorithm. Paper [28] introduced an ensemble method comprising a whale optimization algorithm (WOA), Lévy flight and DE for job shop scheduling problems. Paper [29] introduced an enhanced adaptive differential evolution algorithm based on an adapted crossover rate value and a dynamic population reduction strategy.

Based on the above motivation, the current study aims to develop a general framework for time series forecasting, a neuro-evolutionary NARX model (NE-NARX). The NE-NARX structure is designed by connecting the feedforward neural network and nonlinear autoregressive network with exogenous inputs (NARX). The weights of NE-NARX are optimized by the modified differential evolution (mDE) algorithm which is modified by a mutation scheme and self-adaptive control parameters. The effectiveness of the proposed NE-NARX model is tested on two well-known benchmark datasets, including the Canadian lynx and the Wolf sunspot. The results of the proposed model are compared to other models, including the classical BP, PSO, DE, JDE, JADE, ADE and ARIMA models. Empirical results show the superiority of the NE-NARX model over other models.

After that, the proposed NE-NARX is applied to accurately hourly energy consumption forecasting. To verify the effectiveness of the proposed method, we make a comparison with other machine learning techniques such as the hybrid residual CNN-LSTM model [30], gated recurrent units (GRU) [30], convolutional neural network (CNN) [30], long short-term memory (LSTM) [30], CNN-LSTM [31], Seq2Seq [32] and multilayered GRU [33].

The content of this paper is presented as follows. Section 2 describes the proposed NE-NARX model. Section 2 also discusses the setup control parameter of the NE-NARX model. Section 3 presents prediction results to prove the efficiency and robustness of the investigated model, and eventually, Sect. 4 contains the conclusions.

2 Methodology

The proposed neuro-evolutionary NARX (NE-NARX) model is designed by combining a feedforward neural network (FNN) structure with a nonlinear autoregressive network with exogenous inputs (NARX) model. The weights of novel NE-NARX are optimally identified by a modified differential evolution (mDE) technique. The block diagram requires to be realized in forecasting illustrated in Fig. 1.

2.1 Proposed NE-NARX structure

The nonlinear autoregressive network with exogenous inputs (NARX) denotes a recurrent dynamic network. The NARX structure relies on the linear ARX model commonly used in prediction. The NARX model is described as

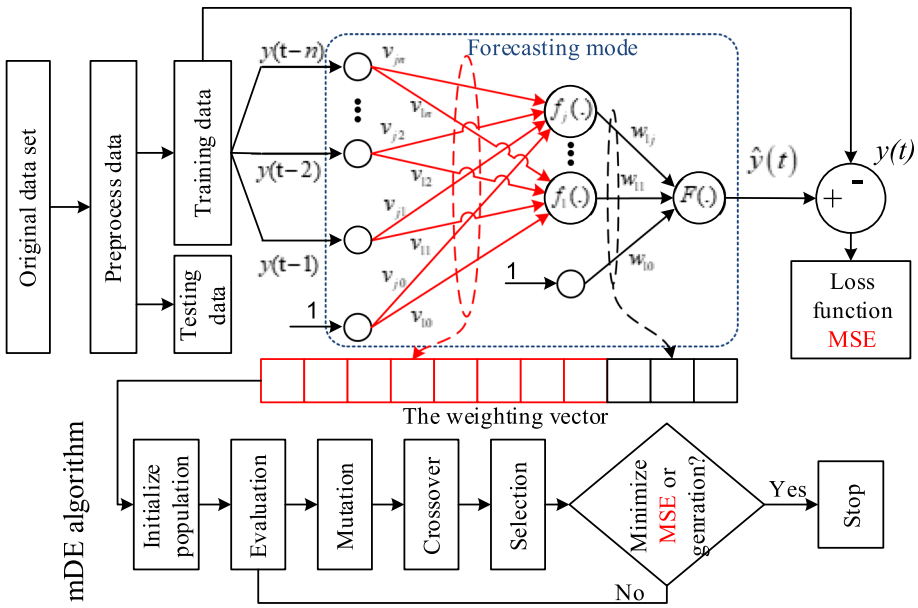

Fig. 1 Proposed NE-NARX model for forecasting

$$y(t) = g(y(t-1), y(t-2), \ldots, y(t-n)) \tag{1}$$

where $y(t-n)$ is an output of preprocessed data block shown in Fig. 1. n represents the backward shift operator as $n^{-1}y\ (t) = y(t-1)$. The preprocessed data block uses the data normalization technique including the logarithms (to the base 10) or Min–Max normalization described in Eq. (2) to transform the values of an original dataset into a common scale.

$$y_{\text{scaled}} = \frac{y - y_{\min}}{y_{\max} - y_{\min}} \tag{2}$$

The next value of the output signal $y(t)$ is regressed on previous values of an observed exogenous response. Then we can implement the NARX model by using a feedforward neural network (FNN) to approximate the function $\boldsymbol{g}$, which is completely described in Fig. 1. Here, v_{jn} denotes the weighting value of the input layer, v_{j0} is the weight of the bias input layer, w_{1j} represents the weighting value of the hidden layer, w_{10} is the weight of the bias hidden layer, $f_j(\cdot)$ denotes a sigmoid function at the hidden layer, and $F(\cdot)$ represents a linear activation function at the output layer, respectively.

$$f(x) = \frac{1}{1+e^{-x}} \quad \text{and} \quad F(x) = x \tag{3}$$

We determine the prediction output as follows:

$$\hat{y}(t,\theta) = F\left(\sum_{i=1}^{j} w_{1i} f_i\left(\sum_{l=1}^{n} v_{il}\varphi_l(k) + v_{l0}\right) + w_{10}\right) \tag{4}$$

where $\varphi(k)$ is the regression vector and θ denotes the weight vector, defined by

$$\varphi(\boldsymbol{t}) = [y(t-1), y(t-2), \ldots, y(t-n)]^T \tag{5}$$

$$\boldsymbol{\theta} = \left[v_{10}, \ldots, v_{1n}, v_{j0}, \ldots, v_{jn}, w_{10}, \ldots, w_{1j}\right]^T = [w_1, \ldots, w_D]^T \tag{6}$$

In case the final model was installed, the learning procedure is realized to fine-tune and evolve the weighting vectors θ of the neural NARX model. These weighting vectors θ should be modified as to approximate the results and to optimize the total residual of the proposed NE-NARX. The error *MSE* designed based on a measure of fitness regarding MSE condition is as follows,

$$E_N(\theta, Z^N) = \frac{1}{2N}\sum_{t=1}^{N}[y(t) - \hat{y}(t|\theta)]^T[y(t) - \hat{y}(t|\theta)] \tag{7}$$

in which the learning data Z^N is described as $Z^N = \{[y(t)]|k = 1, \ldots, N\}$. In this paper, the purpose is to minimally optimize the fitness function E_N in Eq. (7) using a modified differential evolution (mDE) algorithm.

2.2 mDE training algorithm

The differential evolution (DE) algorithm is first studied by Storn and Price [17]. Today, it is becoming a well-known and powerful stochastic swarm-based optimization technique. Since its first publication, the DE technique's variants were studied by modified components such as mutation and crossover procedures, control parameters (i.e., mutant parameter F, crossover parameter CR) or hybrid with other metaheuristic algorithms. In this section, an mDE algorithm is proposed to train the proposed NE-NARX model. In the mDE training algorithm, a modified mutation scheme and self-adaptive control parameters are

focused on research. The mDE technique consists of four following principal steps as,

a. Initialization To minimize the objective function in Eq. (7), at first, the population number NP is chosen. It provides a randomly initialized population with NP D-dimensional vectors as,

$$\boldsymbol{\theta}_{i,G} = \left[w_{1,i,G}, w_{2,i,G}, \ldots, w_{D,i,G}\right], \quad i = 1, \ldots, \mathrm{NP} \tag{8}$$

where D denotes the sum of weighting values and thresholds of the NE-NARX model, i denotes an index to the population and G represents an index to the number generation, $G = 1, 2, \ldots, G_{\max}$. All coefficient vector is limited to a certain searching region which is defined as

$$w_{i,j} = x_{\min,j} + rand_{i,j}[0, 1]\left(w_{\max,j} - w_{\min,j}\right) \tag{9}$$

in which $0 \le rand_{\mathrm{j,i}}[0, 1] \le 1$, $j = 1, 2, \ldots, D$.

b. Mutation The mutant phase is described as the most crucial parameter which affects the capability of the DE algorithm. Then, there are vast differences of mutant policies investigated such as,

1	DE/best/1	$\boldsymbol{v}_{i,G} = \boldsymbol{\theta}_{best,G} + F\left(\boldsymbol{\theta}_{r_1^i,G} - \boldsymbol{\theta}_{r_2^i,G}\right)$
2	DE/rand/1	$\boldsymbol{v}_{i,G} = \boldsymbol{\theta}_{r_1^i,G} + F\left(\boldsymbol{\theta}_{r_2^i,G} - \boldsymbol{\theta}_{r_3^i,G}\right)$
3	DE/best/2	$\boldsymbol{v}_{i,G} = \boldsymbol{\theta}_{best,G} + F\left(\boldsymbol{\theta}_{r_1^i,G} + \boldsymbol{\theta}_{r_2^i,G} - \boldsymbol{\theta}_{r_3^i,G} - \boldsymbol{\theta}_{r_4^i,G}\right)$
4	DE/rand/2	$\boldsymbol{v}_{i,G} = \boldsymbol{\theta}_{r_1^i,G} + F\left(\boldsymbol{\theta}_{r_2^i,G} + \boldsymbol{\theta}_{r_3^i,G} - \boldsymbol{\theta}_{r_4^i,G} - \boldsymbol{\theta}_{r_5^i,G}\right)$
5	DE/current-to-best/1	$\boldsymbol{v}_{i,G} = \boldsymbol{\theta}_{i,G} + F\left(\boldsymbol{\theta}_{best,G} - \boldsymbol{\theta}_{i,G}\right) + F\left(\boldsymbol{\theta}_{r_1^i,G} - \boldsymbol{\theta}_{r_2^i,G}\right)$

Here, $r_1^i, r_2^i, r_3^i, r_4^i$ and r_1^5 are values selected at random within the boundary [1, NP] with $r_1^i \neq r_2^i \neq r_3^i \neq r_4^i \neq r_5^i \neq i$; the scaled coefficient F denotes an arbitrarily chosen value from [0,1].

However, each mutant policies in a classical DE have different strengths and weaknesses, which leads to inconvenient features including sluggishly convergent velocity and easy to be fallen into a local optimal result. For example, paper [34] demonstrates that "DE/best/1," "DE/best/2" and "DE/current-to-best/1" often have a fast convergence rate during solving uni-modal problems. Nonetheless, they are easily fallen into a local optimum solution when solving multimodal problems. In contrast, "DE/rand/1" and "DE/rand/2" possess strong exploration capability in the global search space but slow convergence rates. Paper [35] proposed the scaling factor to make the expected mutation vector equal to DE/best/1 or DE/rand/1. Paper [36] proposed to combine three mutation operators including DE/rand/1, DE/current-to-rand/1 and DE/current-to-pbest/1. Therefore, to balance the exploring global search ability and the locally exploiting ability, a modified mutant policy is designed by comprising "DE/rand/1," "DE/current-to-best/1" and "DE/best/2." In each generation, one mutant policy will be chosen using an adaptive process described as,

Algorithm 1. Mutant phase

if (rand[0,1] < *0.5*) *% using "rand/1"*

$$\boldsymbol{v}_{i,G} = \boldsymbol{\theta}_{r_1^i,G} + F\left(\boldsymbol{\theta}_{r_2^i,G} - \boldsymbol{\theta}_{r_3^i,G}\right)$$

elseif (0.5 ≤ rand[0,1] ≤ *0.75*) *% using "current-to-best /1"*

$$\boldsymbol{v}_{i,G} = \boldsymbol{\theta}_{i,G} + F\left(\boldsymbol{\theta}_{best,G} - \boldsymbol{\theta}_{i,G}\right) + F\left(\boldsymbol{\theta}_{r_1^i,G} - \boldsymbol{\theta}_{r_2^i,G}\right)$$

elseif (0.75 < rand[0,1]) *% using "best/2"*

$$\boldsymbol{v}_{i,G} = \boldsymbol{\theta}_{r_1^i,G} + F\left(\boldsymbol{\theta}_{r_2^i,G} + \boldsymbol{\theta}_{r_3^i,G} - \boldsymbol{\theta}_{r_4^i,G} - \boldsymbol{\theta}_{r_5^i,G}\right)$$

end

c. Crossover In case mutant operation is completed, a crossover operation is performed on the individuals. The mDE algorithm often uses the *binôme* relation which is represented as,

$$u_{j,i,G} = \begin{cases} v_{j,i,G} & \text{if } \left(rand_{j,i}[0, 1] \le \mathrm{CR}\right) \\ w_{j,i,G} & \text{otherwise} \end{cases} \tag{10}$$

with $i = 1, 2, \ldots, NP, j = 1, 2, \ldots, D$. Remark that $rand_{j,i}[0, 1]$ is a uniquely random value, and CR represents the crossover coefficient.

d. Selection The eventual vector $\boldsymbol{\theta}_{i,G}$ is compared with the trial one $\boldsymbol{u}_{i,G}$. This vector contains minor values that will be lived through to the followed generation. The process of choice is described as

$$\boldsymbol{\theta}_{i,G+1} = \begin{cases} \boldsymbol{u}_{i,G} & \text{if } E_N\left(\boldsymbol{u}_{i,G}\right) \le E_N\left(\boldsymbol{\theta}_{i,G}\right) \\ \boldsymbol{\theta}_{i,G} & \text{otherwise} \end{cases} \tag{11}$$

in which $E_N(\boldsymbol{\theta})$ is minimum optimized cost function and is shown in Eq. (7).

2.3 Implementation of control coefficients

The control coefficients of the NE-NARX model are weights values θ, CR, F and NP which importantly influence the efficiency and quality of the proposed model in prediction.

For CR, F and NP, paper [17] suggested that F will be 0.5 as the best choice, CR is set to 0.1 or 0.9, and NP is within $5D$ and $10D$. Paper [37] introduced that NP is within $3D$ and $8D$, CR is within [0.3, 0.9] and $F = 0.6$. Recently, several authors have suggested adaptively/self-adaptively

modifying control coefficients to tune crossover and mutant factors using fuzzy logic [38] and self-adaptive control coefficients by means of evolution (called jDE) in [39]. Paper [40] proposed a mutation operator "DE/current-to-pbest" and adaptive control parameters F- and CR-based Cauchy distribution (called JADE). Using the above-mentioned analysis, the fact is that it is critical to choose control coefficients F, CR and NP. In this paper, it is important to note that coefficients F and CR were not fixed as in the classic DE algorithm, and those two values are stochastically provided by a normal distribution $N(0.7, 0.3)$ with mean value 0.7, standard deviation 0.3 and $N(0.85, 0.15)$, respectively. Such implementation is consulted from [41], whose purpose is to improve searchability in numerous seeking directions. Eventually NP coefficient is not varied while mDE is in operation.

Weights value $\boldsymbol{\theta}$ of NE-NARX model are often initialized within a small limited range around zero, and the parameter bounds are rarely set, or boundary conditions were set very wide. However, paper [42] proved that if input and output data were standardized to [− 1,1], the weights value of the neural network should be initialized within a small range [− 1,1]. Therefore, in this paper, the weights are initialized, and the search space is bounded within [− 1,1].

The full description of the mDE technique is presented as Algorithm-2. Here, *Max-MSE* denotes allowable error; $G_{\max}$ denotes the maximum value of iterations; and *randint* (1, D) represents a function showing a uniform stochastic value within 1 and D.

2.4 Evaluation metrics

The proposed model is evaluated based on the precision of the prediction. Typical forecast precision measures such as MSE (mean square error) and MAPE (mean absolute percent error) were criticized due to their instability in the case of changing numbers of test sample predicting periods. This paper proposes two adopted precision measures described as:

$$\mathrm{MSE} = \frac{1}{2N}\sum_{t=1}^{N}[y(t) - \hat{y}(t)]^T[y(t) - \hat{y}(t)] \tag{12}$$

$$\mathrm{MAPE} = \frac{100}{N}\sum_{t=1}^{N}\left|\frac{y(t) - \hat{y}(t)}{y(t)}\right| \tag{13}$$

Here, $\boldsymbol{y(t)}$ denotes the actual value at time t, $\hat{y}(t)$ shows the predicted value at time t, and N denotes the total number of the observed response.

3 Results and discussions

3.1 The benchmark test

The effectiveness of the NE-NARX model is tested on two well-known datasets, including the Canadian lynx data and the Wolf sunspot data, in which the lynx dataset used in time series forecasting contains the number of lynx trapped per year in the Mackenzie River district of North-West Canada. The Wolf Sunspot data series are recording the

Algorithm 2. Pseudocode of mDE algorithm

```
Generate the initial population of size NP
Generate the initial weighting value θ of NE-NARX model
while (Max-MSE or Gmax is not reached) do
    for i = 1 to NP do
        CR = randn [0.85, 0.15]
        F  = randn [0.7, 0.3]
        jrand = randint(1, D)
        for j = 1 to D do
            if rand[0,1] < CR or j = jrand then
                Algorithm 1. Mutant phase
            else
                u_{i,j} = w_{i,j}
            end
        end
        if E_N(u_i) ≤ E_N(θ_i)
            θ_i = u_i
        else
            θ_i = θ_i
        end
    end
end
```

Table 1 Algorithm's parameters applied in the estimated procedure

Algorithm	Parameters	Notice
Generalized	Hidden node number	S1
	Population size	NP
	Number of iterations	G_max
	Bounds of weighting magnitudes	[− 1,1]
BP	Learning ratio, η	0.0001
	Number of iterations	5000
PSO	Inertial weighting value, w	0.8
	Element's best value, $c1$	2
	Population's best value, $c2$	2
DE [17]	Mutation coefficient, F	0.5
	Crossover coefficient, CR	0.9
jDE [39]	$F_{i,G+1} = \begin{cases} F_l + rand_1 * F_u, & \text{if } rand_2 < \tau_1 \\ F_{i,G}, & \text{otherwise} \end{cases}$	$\tau_1 = \tau_2 = 0.1, F_l = 0.1$
	$CR_{i,G+1} = \begin{cases} rand_3, & \text{if } rand_4 < \tau_2 \\ CR_{i,G}, & \text{otherwise} \end{cases}$	F_u = 0.9
JADE [40]	$CR_i = randn_i(\mu_{CR}, 0.1)$	S_{CR}: set of all successful crossover factors; S_F: set of all successful mutation factors
	$\mu_{CR} = (1-c)\mu_{CR} + c * \text{mean}_A(S_{CR})$	
	$F_i = randc_i(\mu_F, 0.1)$	
	$\mu_F = (1-c)\mu_F + c * \text{mean}_L(S_F)$	μ_{CR} and μ_F: initialized to be 0.5
ADE [43]	$CR = \max\{0, 7, 1 - f(G_b)/f(W_b)\}$	G_b: the current global optimal vector
	$F = rand * 0.1 + 0.45$	W_b: the global worst vector

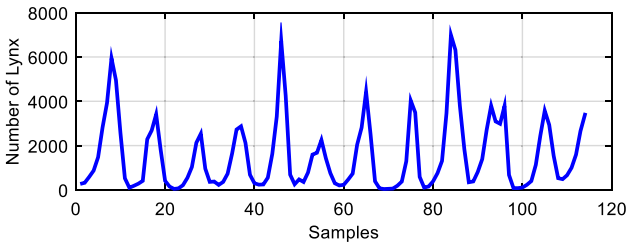

Fig. 2 Canadian lynx data series (1821–1934)

annual sunspot indices from 1700 to 1999. To verify the superiority of mDE technique, we make comparisons with BP, PSO, DE, ADE, JDE and JADE approaches. These algorithms are realized with MATLAB 2015b on Intel Core i5, a velocity of 2.53 GHz and 8 GB RAM. Each algorithm runs 20 times. Table 1 fully presents the algorithm's parameters used in the forecasting procedure.

3.1.1 Case study A: Canadian lynx data

At first, the Canadian lynx dataset is a plot in Fig. 2, which shows a periodicity of approximately ten years, with 114 observations from the period of 1821–1934. The dataset is normalized by the logarithms (to the base 10) and is

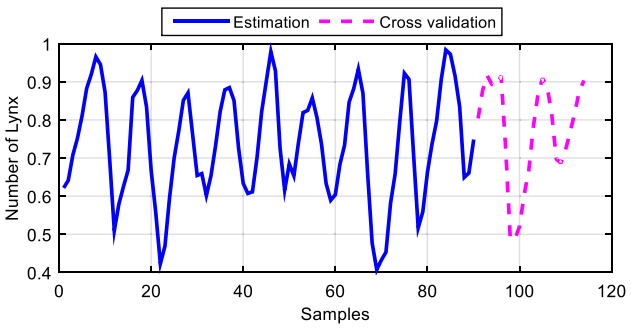

Fig. 3 Canadian lynx data normalization for estimating and validating

partitioned into 80% of the dataset as an estimating set and 20% of the dataset as a validating set as presented in Fig. 3.

Secondly, the NE-NARX model is chosen by combining the three-layered FNN composed of S1 hidden nodes and the fourth-order NARX model. Finally, the estimating and validating processes are realized to identify the NE-NARX structure. Choose S1 = 8, NP = 100, and G_max = 5000.

Thirdly, the prediction of performance through the proposed NE-NARX is made. To survey the effectiveness of the NE-NARX model, which is compared to the BP, PSO, DE, JDE, JADE and ADE models, respectively. The coefficients of the BP, PSO, DE, JDE, JADE and ADE algorithms are fully presented in Table 1. An ARIMA

Table 2 Performance comparisons in prediction (Case A)

Method	MSE					MAPE %
	Estimating				Validating	
	Best	Worst	Average	SD	Average	Average
NE-NARX	**0.0011**	**0.0012**	**0.0012**	**3.7857e−05**	**0.0014**	**5.1833**
ADE-NARX	0.0017	0.0019	0.0018	7.5530e−05	0.0016	5.8050
JADE-NARX	0.0014	0.0015	0.0014	3.3211e−05	0.0015	5.9680
JDE-NARX	0.0016	0.0022	0.0018	1.6199e−04	0.0016	6.4618
DE-NARX	0.0020	0.0031	0.0027	3.5375e−04	0.0019	6.8795
PSO-NARX	0.0021	0.0042	0.0033	6.1337e−04	0.0022	8.2442
BP-NARX	0.0021	0.0075	0.0041	0.0042	0.0039	11.3156
ARIMA (1,1,1)	0.0069				0.0043	7.9655
ARIMA (2,1,1)	0.0047				0.0036	7.1449

Bold represents the best value in each column

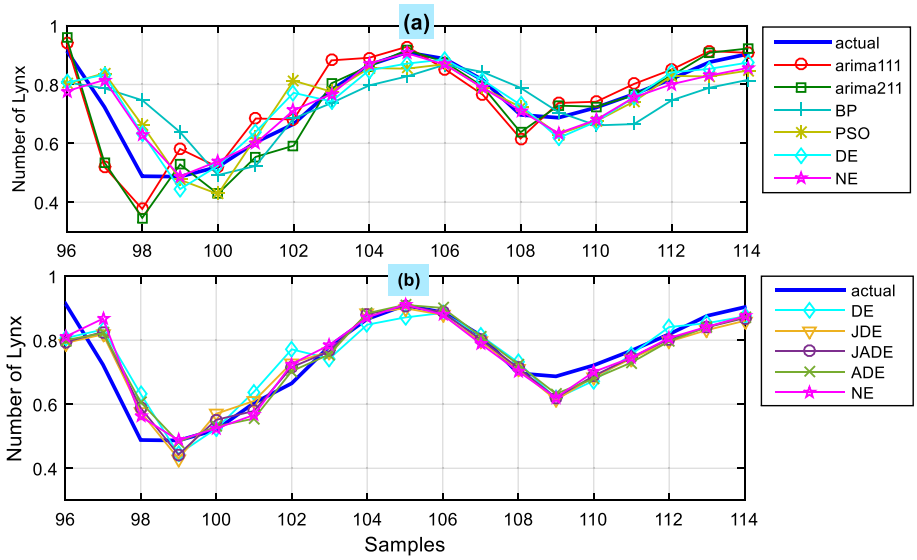

Fig. 4 Performance comparison in prediction

(1,1,1) and ARIMA (2,1,1) model based on the ARIMA function of MATLAB are used as the benchmark for evaluating the capacity of the proposed NE-NARX model. Table 2 gives the comparative performance of all models on the estimating and validating processes. Figure 4 illustrates the performance forecasting in the validating procedure. Here, Fig. 5a focuses on comparing NE-NARX with several classical methods and Fig. 5b depicts the comparison of DE's variants.

Analyzing the above-mentioned results, it is clear to note that the estimating and cross-validating errors of the proposed NE-NARX confirm that it obtains precise results and provides better results in comparison with ADE,

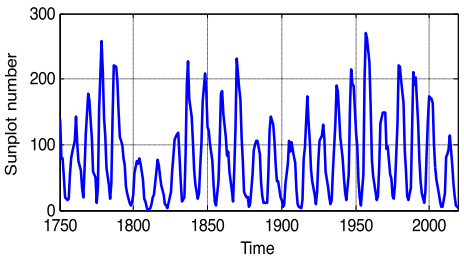

Fig. 5 Yearly mean total sunspot number [1750–2019]

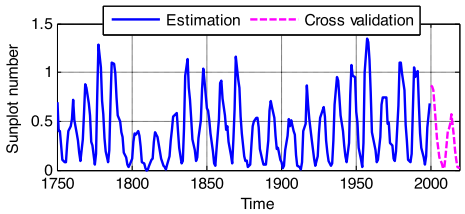

Fig. 6 Sunspot number normalization for estimating and validating

Table 3 Comparative results in prediction (Case B)

Method	MSE					MAPE %
	Estimating				Validating	
	Best	Worst	Average	Std. Dev	Average	Average
NE-NARX	**0.0046**	**0.0048**	**0.0047**	**7.6863e−05**	**0.0048**	**11.49**
ADE-NARX	0.0076	0.0103	0.0087	7.5133e−04	0.0065	12.18
JADE-NARX	0.0055	0.0059	0.0054	1.5137e−04	0.0064	11.71
JDE-NARX	0.0074	0.0085	0.0079	3.5887e−04	0.0063	11.60
DE-NARX	0.0088	0.0118	0.0095	4.1622e–04	0.0062	12.43
PSO-NARX	0.0084	0.0111	0.0099	8.35e−4	0.0071	12.59
BP-NARX	0.0093	0.0139	0.0119	0.0014	0.0077	13.68
ARIMA (4,1,1)	0.0193				0.0111	17.52
ARIMA (2,1,1)	0.0212				0.0142	18.15
ARIMA (2,1,2)	0.0183				0.0109	15.08

Bold represents the best value in each column

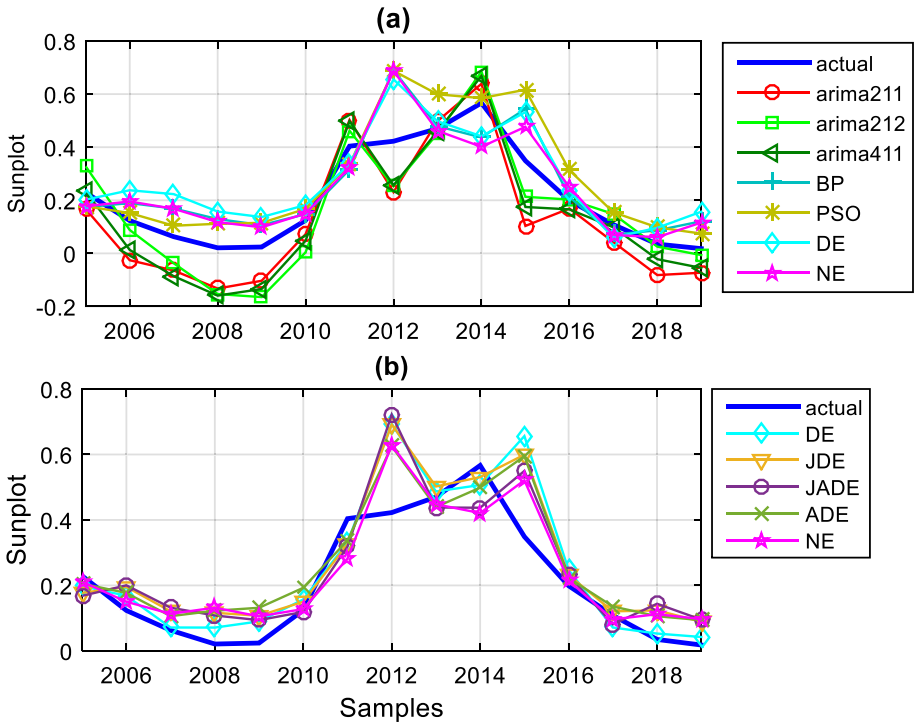

Fig. 7 Performance comparison in prediction (Case B)

Table 4 Statistics of PJME hourly energy consumption

Dataset	PJME dataset	PJME normalization
Sample size	145,366	145,366
Mean	3.2080e+04 MW	0.3695
Std	6.464e+03 MW	0.1362
Min	1.4544e+04 MW	0
Max	6.2009e+04 MW	1

JADE, JDE, DE, PSO and BP approaches. In particular, the proposed NE-NARX model gives superior results compared to ARIMA (1,1,1) and ARIMA (2,1,1) models. Also, it should be noted that the MSE and MAPE with the proposed NE-NARX model are the lowest ones among the comparative models, followed by NE-NARX with MSE of 0.0011 and MAPE of 5.1833, ADE with MSE of 0.0016 and MAPE of 5.8050, JADE with MSE of 0.0015 and MAPE of 5.9680, JDE with MSE as 0.0016 and MAPE as 6.4618, DE with MSE of 0.0019 and MAPE of 6.8795, PSO with MSE of 0.0022 and MAPE of 8.2442, BP with MSE of 0.0039 and MAPE of 11.3156, and ARIMA (2,1,1) with MSE of 0.0036 and MAPE of 7.1449.

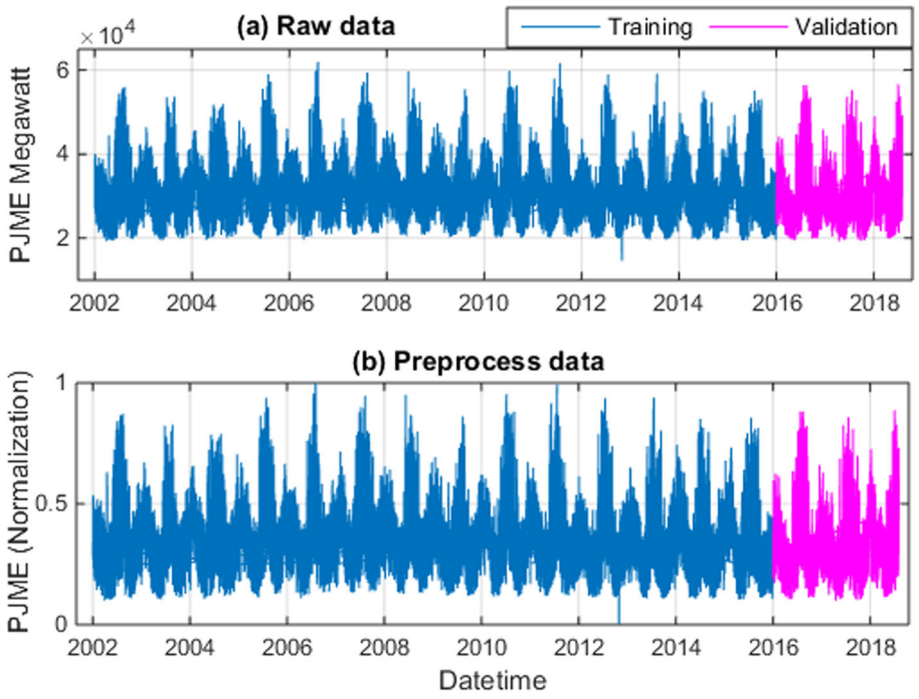

Fig. 8 PJME hourly energy consumption [2002–2018]

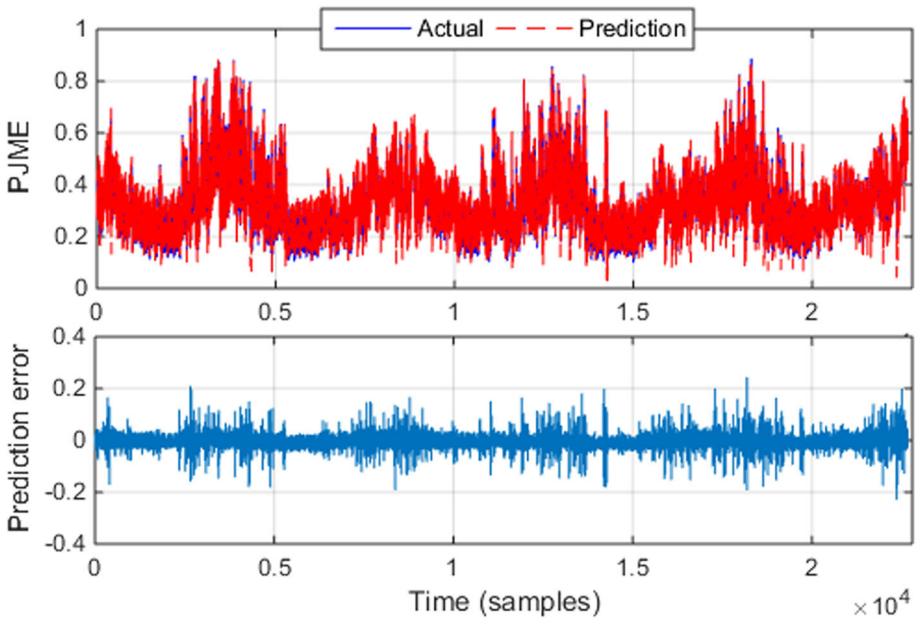

Fig. 9 Performance prediction on PJME validation set

3.1.2 Case study B: Wolf sunspot data

At first, the Wolf sunspot dataset is a plot in Fig. 5, which shows the annual sunspot indices from 1750 to 2019 (collected via the link http://www.sidc.be/silso/datafiles). The dataset is normalized and is partitioned into 85% of the dataset as an estimating set and 15% of the dataset as a validating set as presented in Fig. 6.

Secondly, the proposed NE-NARX model is chosen by combining the three-layered FNN composing of S1 hidden nodes, the fifth-order NARX model. Finally, the estimating and validating processes are realized to identify the NE-NARX structure. Choose S1 = 11, NP = 100, and G_max = 5000.

Thirdly, the prediction of performance through the proposed NE-NARX is made. To survey the effectiveness of the NE-NARX model, it is compared with the BP, PSO, DE, JDE, JADE and ADE models. The coefficients of the BP, DE, JDE, JADE, ADE and PSO approaches are illustrated in Table 1. An ARIMA (2,1,1), ARIMA (4,1,1) and ARIMA (2,1,2) models are used as the benchmark for evaluating the capacity of the proposed NE-NARX

Table 5 Performance comparison on PJME

Methods	MSE	MAPE %
NE-NARX	2.34e−4	4.86
CNN-LSTM [30]	0.002	5.82
CNN [30]	0.194	43.92
LSTM [30]	0.103	37.53
GRU [30]	0.265	47.48
CNN-LSTM [31]	0.3549	32.83
Seq2Seq [32]	0.3906	–
Multilayered GRU [33]	0.0081	–

structure. Table 3 gives the comparative performance of all models on the estimating and validating process. Figure 7 illustrates the performance forecasting in the validating procedure.

Analyzing the above-mentioned results, it is clear to see that the estimating and cross-validating errors of the NE-NARX once more confirm that it obtains strongly precise results and generates better-identified values in comparison with ones of ADE, JDE, JADE, DE, PSO and BP models. In particular, three ARIMA models archive the worst performance in terms of the MSE when compared with other BP, PSO, DE, JDE, ADE, JADE and NE-NARX models. Also, it should be noted that the proposed NE-NARX model is the lowest among the nine models with the MSE and MAPE values of 0.0046 and 11.49, respectively.

3.2 Hourly energy consumption prediction

The hourly power consumption dataset is collected from PJM Interconnection website which has data from 2002 to 2018 for the entire eastern region (called PJME dataset). Table 4 shows the characteristics of PJM hourly energy consumption dataset. The data are normalized by using min–max feature scaling and are split into estimating and validating sets in which the data after 2015 are used as validating set, as presented in Fig. 8, in which Fig. 8a shows the PJME raw data and Fig. 8b shows the preprocessed data.

Secondly, the NE-NARX model is selected by combining the three-layered FNN composed of S1 hidden nodes and the fourth-order NE-NARX model. The estimating and validating processes are realized to identify the NE-NARX structure. Choose S1 = 10, NP = 100, and G_max = 2000. Thirdly, the prediction of performance through the proposed NE-NARX is made. Figure 9 shows performance prediction time series data for PJM hourly energy.

The performance of NE-NARX model is compared with a hybrid residual CNN-LSTM model [30], gated recurrent units (GRU) [30], convolutional neural network (CNN) [30], long short-term memory (LSTM) [30], CNN-LSTM [31], Seq2Seq [32] and multilayered GRU [33], which are shown in Table 5. It was found that the NE-NARX model achieved 2.34e−4 MSE and 4.86% MAPE, which are lower errors than the other methods. Therefore, it can be seen that the NE-NARX model has good forecasting performance for PJME hourly energy consumption.

4 Conclusions

In this paper, we have applied an advanced neuro-evolutionary NARX or NE-NARX model for time series forecasting. NE-NARX model is an ensemble methodology comprising feedforward neural networks, modified differential evolution (mDE) algorithm and a nonlinear autoregressive network with exogenous inputs (NARX). To evaluate the effectiveness of the NE-NARX model, we compared it to those of the well-known models including BP, PSO, DE, JDE, ADE, JADE and ARIMA models. And then, NE-NARX model is used for hourly energy consumption forecasting and is compared with other machine learning techniques including GRU, CNN, LSTM, CNN-LSTM, sequence-to-sequence learning and multilayered GRU. The empirical results with three actual datasets proved that the proposed NE-NARX model archives the best performance measure based on MAPE and MSE criteria among all the models.

Data availability The dataset used in this research is included in this paper.

Declarations

Conflict of interest The authors declare no conflict of interest.

References

1. Guo J, He H, Sun C (2019) ARIMA-based road gradient and vehicle velocity prediction for hybrid electric vehicle energy management. IEEE Trans Veh Technol 68(6):5309–5320
2. Singh SN, Mohapatra A (2019) Repeated wavelet transform based ARIMA model for very short-term wind speed forecasting. Renew energy 136:758–768
3. de Oliveira EM, Oliveira FLC (2018) Forecasting mid-long term electric energy consumption through bagging ARIMA and exponential smoothing methods. Energy 144:776–788
4. Xu Y, Li F, Asgari A (2022) Prediction and optimization of heating and cooling loads in a residential building based on multi-layer perceptron neural network and different optimization algorithms. Energy 240:122692
5. Lei R, Yin J (2022) Prediction method of energy consumption for high building based on LMBP neural network. Energy Rep 8:1236–1248

6. Saâdaoui F (2017) A seasonal feedforward neural network to forecast electricity prices. Neural Comput Appl 28(4):835–847
7. Hong Y-Y, Satriani TRA (2020) Day-ahead spatiotemporal wind speed forecasting using robust design-based deep learning neural network. Energy 209:118441
8. Huang C et al (2020) Robust forecasting of river-flow based on convolutional neural network. IEEE Trans Sustain Comput 5(4):594–600
9. Bandara K, Bergmeir C, Smyl S (2020) Forecasting across time series databases using recurrent neural networks on groups of similar series: a clustering approach. Expert Syst Appl 140:112896
10. Hewamalage H, Bergmeir C, Bandara K (2021) Recurrent neural networks for time series forecasting: current status and future directions. Int J Forecast 37(1):388–427
11. Chen M-R, Zeng G-Q, Lu K-D, Weng J (2019) A two-layer nonlinear combination method for short-term wind speed prediction based on ELM, ENN, and LSTM. IEEE Internet Things J 6(4):6997–7010
12. Zhang Z, Ding S, Sun Y (2020) A support vector regression model hybridized with chaotic krill herd algorithm and empirical mode decomposition for regression task. Neurocomputing 410:185–201
13. Feng Z, Niu W (2021) Hybrid artificial neural network and cooperation search algorithm for nonlinear river flow time series forecasting in humid and semi-humid regions. Knowl Based Syst 211:106580
14. Talaat M, Farahat MA, Mansour N, Hatata AY (2020) Load forecasting based on grasshopper optimization and a multilayer feed-forward neural network using regressive approach. Energy 196:117087
15. Cai W, Yang J, Yu Y, Song Y, Zhou T, Qin J (2020) PSO-ELM: A hybrid learning model for short-term traffic flow forecasting. IEEE Access 8:6505–6514
16. Kılıç F, Yılmaz İH, Kaya Ö (2021) Adaptive co-optimization of artificial neural networks using evolutionary algorithm for global radiation forecasting. Renew Energy 171:176–190
17. Storn R, Price K (1995) Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces, vol 3. ICSI Berkeley
18. Liang J et al (2020) Parameters estimation of solar photovoltaic models via a self-adaptive ensemble-based differential evolution. Sol Energy 207:336–346
19. Song Y et al (2021) MPPCEDE: multi-population parallel co-evolutionary differential evolution for parameter optimization. Energy Convers Manag 228:113661
20. Wang Y, Liu H, Long H, Zhang Z, Yang S (2017) Differential evolution with a new encoding mechanism for optimizing wind farm layout. IEEE Trans Ind Inform 14(3):1040–1054
21. Wang L, Hu H, Ai X-Y, Liu H (2018) Effective electricity energy consumption forecasting using echo state network improved by differential evolution algorithm. Energy 153:801–815
22. Peng L, Liu S, Liu R, Wang L (2018) Effective long short-term memory with differential evolution algorithm for electricity price prediction. Energy 162:1301–1314
23. Deng W, Liu H, Xu J, Zhao H, Song Y (2020) An improved quantum-inspired differential evolution algorithm for deep belief network. IEEE Trans Instrum Meas 6:66
24. Zhang Q, Shen X, Zhao J, Xiao Q, Huang J, Wang Y (2020) Hysteresis modeling of piezoelectric actuator using particle swarm optimization-based neural network. Proc Inst Mech Eng Part C J Mech Eng Sci 234(23):4695–4707
25. Van Kien C, Anh HPH, Son NN (2021) Adaptive inverse multilayer fuzzy control for uncertain nonlinear system optimizing with differential evolution algorithm. Appl Intell 51(1):527–548
26. Jamali A, Mallipeddi R, Salehpour M, Bagheri A (2020) Multi-objective differential evolution algorithm with fuzzy inference-based adaptive mutation factor for Pareto optimum design of suspension system. Swarm Evol Comput 54:100666
27. Zhang Z, Ding S, Jia W (2019) A hybrid optimization algorithm based on cuckoo search and differential evolution for solving constrained engineering problems. Eng Appl Artif Intell 85:254–268
28. Liu M, Yao X, Li Y (2020) Hybrid whale optimization algorithm enhanced with Lévy flight and differential evolution for job shop scheduling problems. Appl Soft Comput 87:105954
29. Li S, Gu Q, Gong W, Ning B (2020) An enhanced adaptive differential evolution algorithm for parameter extraction of photovoltaic models. Energy Convers Manag 205:112443
30. Khan ZA et al (2022) Efficient short-term electricity load forecasting for effective energy management. Sustain Energy Technol Assess 53:102337
31. Kim T-Y, Cho S-B (2019) Predicting residential energy consumption using CNN-LSTM neural networks. Energy 182:72–81
32. Marino DL, Amarasinghe K, Manic M (2016) Building energy load forecasting using deep neural networks. In: IECON 2016—42nd annual conference of the IEEE industrial electronics society, pp 7046–7051
33. Han T, Muhammad K, Hussain T, Lloret J, Baik SW (2020) An efficient deep learning framework for intelligent energy management in IoT networks. IEEE Internet Things J 8(5):3170–3179
34. Qin AK, Huang VL, Suganthan PN (2009) Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans Evol Comput 13(2):398–417. https://doi.org/10.1109/TEVC.2008.927706
35. Opara K, Arabas J (2018) Comparison of mutation strategies in differential evolution—a probabilistic perspective. Swarm Evol Comput 39:53–69
36. Wu G, Mallipeddi R, Suganthan PN, Wang R, Chen H (2016) Differential evolution with multi-population based ensemble of mutation strategies. Inf Sci 329:329–345
37. Gämperle R, Müller SD, Koumoutsakos P (2002) A parameter study for differential evolution. Adv Intell Syst fuzzy Syst Evol Comput 10:293–298
38. Liu J, Lampinen J (2005) A fuzzy adaptive differential evolution algorithm. Soft Comput 9(6):448–462
39. Brest J, Bošković B, Greiner S, Žumer V, Maučec MS (2007) Performance comparison of self-adaptive and adaptive differential evolution algorithms. Soft Comput 11(7):617–629
40. Zhang J, Sanderson AC (2009) JADE: adaptive differential evolution with optional external archive. IEEE Trans Evol Comput 13(5):945–958
41. Wang Y, Cai Z, Zhang Q (2011) Differential evolution with composite trial vector generation strategies and control parameters. Evol Comput IEEE Trans 15(1):55–66
42. Piotrowski AP (2014) Differential evolution algorithms applied to neural network training suffer from stagnation. Appl Soft Comput 21:382–406
43. Peng Y, He S, Sun K (2022) Parameter identification for discrete memristive chaotic map using adaptive differential evolution algorithm. Nonlinear Dyn 107(1):1263–1275