IET Generation, Transmission & Distribution

*Research Article*

IET Journals
The Institution of Engineering and Technology

# Mid-term electricity load forecasting by a new composite method based on optimal learning MLP algorithm

ISSN 1751-8687
Received on 30th June 2019
Revised 13th October 2019
Accepted on 14th November 2019
E-First on 30th January 2020
doi: 10.1049/iet-gtd.2019.0797
www.ietdl.org

*Mostafa Askari*[1], *Farshid Keynia*[1] ✉

[1]*Department of Energy Management and Optimization, Institute of Science and High Technology and Environmental Sciences, Graduate University of Advanced Technology, Kerman, Iran*

✉ *E-mail: f.keynia@kgut.ac.ir*

**Abstract:** Electricity load forecasting has been developed as an important issue in the deregulated power system in recent years. Many researchers have been working on the prediction of daily peak load for next month as an important type of mid-term load forecasting (MTLF). Nowadays, MTLF provides useful information for assessing environmental impacts, maintenance scheduling, adequacy assessment, scheduling of fuel supplies and limited energy resources etc. The characteristics of mid-term load signal, such as its non-stationary, volatile and non-linear behaviour, present serious challenges for this forecasting. On the other hand, many input variables and relative parameters can affect the load pattern. In this study, a new composite method based on a multi-layer perceptron neural network and optimisation techniques has been proposed to solve the MTLF problem. The proposed method has an optimal training algorithm composed of two search algorithms (particle swarm optimisation and improved ant lion optimiser) and a multi-layer perceptron neural network. The accuracy of the proposed forecast method is extensively evaluated based on several benchmark datasets.

## 1 Introduction

The first paper on load forecasting was published in 1966. Today, load forecasting is an important factor in planning, not only for power systems but also for all companies whose interests depend on the accuracy of the load forecast [1].

On the basis of the time-scale, the load forecast in power systems is generally divided into three types:

### *1.1 Short-term load forecasting (STLF)*

In STLF, forecast horizon and forecast steps are daily and hourly, respectively. Two of the most important applications of this case are unit commitment and economic dispatch.

### *1.2 Mid-term load forecasting (MTLF)*

In MTLF, forecast horizon and forecast steps are monthly and daily, respectively. Two of the most important applications of this case are hydro thermal coordination and maintenance scheduling.

### *1.3 Long-term load forecasting (LTLF)*

In LTLF, forecast horizon and forecast steps are yearly and weekly/monthly, respectively. One of the most important applications of this case is system expansion planning.

The basis of the performance of the different load forecasting methods is the same so that the process of load-affected parameters has been investigated in the past and the future trend is forecasted [1, 2].

The kind of load forecasting technique has been investigated in previous papers. For example, Khuntia *et al*. [1] reviewed the different mid-term and LTLF techniques for electrical power systems. An autonomous approach with load modelling for prediction of daily peak load for the next month is proposed in [2]. The application of artificial neural networks (ANNs) in STLF is reviewed in [3]. In addition, so far in MTLF and LTLF, ANNs are accepted [4]. The use of a support vector machine in load forecasting is reported in [5–10]. A dynamic ANN based on self-learning at each layer proposed in [11]. In [12], a model based on the autonomous approach is proposed. Also, the results of the statistical method with stochastic to forecasting are compared. In [13], a short-term load forecasting method with a new input selection is proposed and to evaluate the performance of the method, the New England load data are used. A comparison of the proposed approach with the other methods shows a significant improvement in forecast accuracy. The different models and methods used to forecast future load demands reviewed in [14]. Adaptive techniques, such as hybrid methods and case-based reasoning models reviewed in [15]. Composite forecast methods with using heuristic optimisation algorithms have been proposed: genetic algorithm in [16, 17], fruit fly optimisation algorithm in [18], particle swarm optimisation (PSO) in [19] and harmony search in [20]. The combination of fuzzy algorithms with other techniques used for forecasts is reported in [21]. In [22], an effective short-term load forecasting model with a feature selection technology and binary genetic algorithm for optimisation is developed that has forecasting results with high accuracy. In [23], an optimum hybrid model is proposed to load forecasting and achieve accurate results rather than other hybrid models. A load forecasting model based on deep residual networks is proposed and multiple test cases are used to prove the effectiveness of the method in [24].

Contributions of this study can be summarised as the following:

i. One of the worst cases that can happen to a power system is the peak load time. If the peak load is accurately forecasted and planned, the highest reliability coefficient is provided for the system. Therefore, in this study, the forecast horizon and forecast steps are monthly and daily (peak load), respectively. In fact, one month from next year is forecasted daily (peak load per day).
ii. The combined method of multi-layer perceptron (MLP) and optimisation algorithms (PSO with adaptive inertia weight and ant lion optimisation (ALO)) are used to MTLF, really the ALO and PSO algorithms are used as a trainer for MLP's parameters based on one year/seasonal peak daily data for training.
iii. Other goals in this study are to determine the optimal data to apply to the neural network as well as to create the optimal structure in the neural network so that the validation error is minimised (for details see Sections 5–6). By optimising the input data while increasing the forecast accuracy, it also

reduces the computational time of the problem. Creating the optimal structure of the neural network is actually the determination of the number of neurons in the middle layer which can be an arbitrary parameter and on the other hand such as this study can be considered as a decision variable in the ALO and PSO optimisation algorithms.

iv. Daily peak load from two datasets are used to prove the effectiveness of the combined method.

The organisation of this study is as follows: the basis of PSO and ALO algorithms and MLP, the forecasting approach, simulation results and discussions, conclusions are presented in Sections 2–7, respectively.

## 2 PSO algorithm

PSO algorithm is a population-based stochastic method. This method is inspired by the social behaviour of fishes, and one of the most important qualities of it is the ability to perform an effective search.

In the following, we described the original PSO algorithm and PSO algorithm with adaptive inertia weight.

### 2.1 Original PSO algorithm

PSO algorithm was first developed in 1995 [25]. The algorithm's performance is such that when determining the initial position for each particle, the value of the fitness function and the velocity corresponding to each particle is calculated, and then these particles move in the search space. In fact, at first, a population of answers will randomly move into the domain of the problem, and the search space will be searched to find the optimal answer. Given the minimum/maximum problem, and based on the limits set for the decision variables, $N_p$ particles (population size) move randomly in the search space, and always store in memory the best value that is obtained by each of them, as well as the best value found between all the particles.

The PSO algorithm contains the following steps:

*Step 1*. The particle position is randomly initialised and, given the number of particles ($N_p$), as well as the number of decision variables ($m$), the resulting matrix represented by $\boldsymbol{x}$ with $N_p$ of the rows and $m$ of the columns.

*Step 2*. As in the previous step, the corresponding velocity of each particle is randomly initialised so that the resulting matrix represented by $\boldsymbol{v}$ with $N_p$ of the rows and $m$ of the columns.

*Step 3*. The best amount of population so far found and the best position ever discovered by each particle are named $gb$ and $pb$, respectively, so that in the first iteration $pb = x$, and after calculating the fitness function corresponding to $x$, if the type of the minimisation problem, $gb = \text{Min } pb$ and if the maximisation type, $gb = \text{Max } pb$.

*Step 4*. By using (1), the velocity of each particle is updated. It should be noted here that $\omega$ is a constant and positive

$$v_i^{t+1} = \omega \times v_i^t + c_1 \times r_1 \times \left(pb_i^t - x_i^t\right) + c_2 \times r_2 \times \left(gb^t - x_i^t\right) \quad (1)$$

inertia weight. In (1), $t$ is the iteration times ($t = 1,2,3, \ldots, t_{max}$), $i = 1,2,3, \ldots, N_p$, $r_1$, and $r_2$ are random numbers with uniform distribution between 0 and 1, $c_1/c_2$ are the individual/social knowledge learning factors, respectively, which in fact control the balance between individual and social knowledge.

*Step 5*. By using (2), the position of each particle is updated again.

*Step 6*. Given the new values from step 4 and step 5, the value of the fitness function is calculated corresponding to the new $x$. The new $pb$ is obtained so that the fitness of each particle in the new $x$ and the previous $pb$ is compared and based on the type of problem, only if the fitness of each particle in the new $x$ is greater than the equivalent of the corresponding particle in the previous $pb$, the particle in the new $pb$ will be replaced. Finally, the fitness of the particles in the new $pb$ is calculated and the new $gb$ is also determined by the type of problem (min/max) from the new $pb$.

*Step 7*. Steps 1–6 are repeated until the stop condition is established.

### 2.2 PSO with adaptive inertia weight

In the PSO algorithm with adaptive inertia weight [26], instead of using a constant value for $\omega$ in (1) an adaptive inertia weight is used, as (3) to balance the local and global search. In (3), $\omega_{min}$ and $\omega_{max}$ are the final and initial inertia weights, respectively. $\omega_{max}$ is the weighting factor at the start of the algorithm and $\omega_{min}$ is the weighting factor at the end of the algorithm, so that at the beginning of the algorithm, the kind of search in the problem space is global and the kind of search is local as $t$ increased

$$x_i^{t+1} = v_i^{t+1} + x_i^t \quad (2)$$

$$\omega^t = \omega_{max} + \frac{\omega_{min} - \omega_{max}}{t_{max}} \times t \quad (3)$$

In this study, the PSO algorithm with adaptive inertia weight is used.

## 3 ALO algorithm

The ALO [27] is one of the nature-inspired methods for solving optimisation problems. This is a population-based metaheuristic algorithm inspired by the interaction between ant and ant lion, the trapping of ant and its hunt by the special method.

The ALO algorithm produces a primitive population of ants and ant lions randomly in the first stage. In other words, in order to solve the problem, a random answer set of ants was created and randomly moved into the search space. In fact, by creating a random initial answer set in the entire search space, the exploration rate in the algorithm will be high. At each stage, while updating the position of the ants, it is also prevented from moving outside of the search space.

Ant lions build a trap proportional to their fitness, so that the higher the fitness of ant lions, the larger traps they will be. When ants fall into an ant lion's trap, an ant lion prevents ants from escaping by throwing stones at the trapping edges. In other words, when ants fall into the trap of an ant lion, the algorithm raises exploitation.

After hunting an ant by ant lion, the position of ant lion should be updated. In other words, in the case of hunting an ant by an ant lion, due to the higher fitness of the ant into the ant lion, the position of the ant lion should be updated with the position of the hunted ant. In addition, in each iteration, the best ant lions, according to fitness, are stored as elite. In the following, the position of the ants is updated according to the average position of the elite and the position of the ant lions, which is determined by the roulette wheel and at each stage, the elite also updates. These steps will continue until the stop condition is met. In the end, the elite is returned as the final answer.

## 4 Multi-layer perceptron (MLP)

ANNs were introduced for the first time in 1943 [28]. The structure of the ANNs is based on the models that humans imagine for the brain and its neurons. The main feature of these networks is its ability to learn the complex relationships between its input and output vectors. So far, different ANNs have been introduced, feed-forward [29], recurrent [30], spiking [31], Kohonen self-organising [32], and radial basis function [33].

The feed-forward neural networks (FNNs) consist of one or more input layers, one or more intermediate or hidden layers and one or more output layers that are placed in parallel. If the FNNs consist of one hidden layer, it is known as a MLP [29]. There are a number of neurons in each of the layers. The input vector only applies to input layer neurons but to the other neurons of the layers, the exits of the neurons of the previous layer are applied. The number of neurons in the input and output layers depends on the number of inputs and outputs of the problem While selecting the

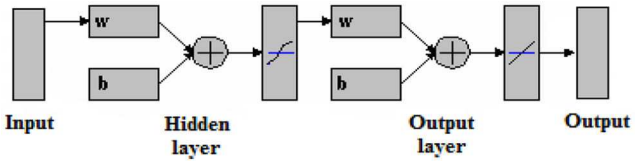


**Fig. 1** *Example of MLP*

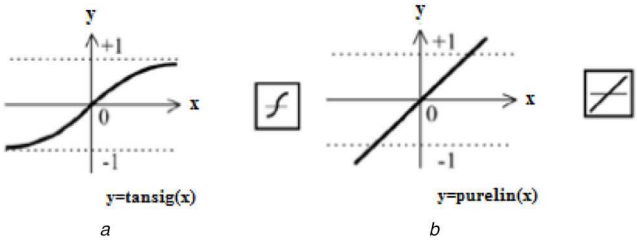


**Fig. 2** *Transfer functions were used*
***(a)*** Tan-sigmoid, ***(b)*** Linear

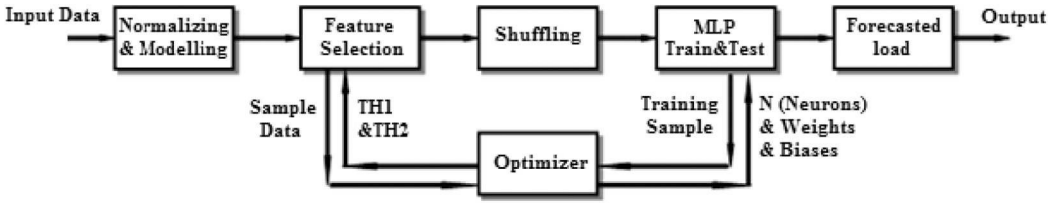


**Fig. 3** *Block diagram of a composite approach used for MTLF*

number of hidden layer neurons is an arbitrary parameter or like this paper obtained by stochastic optimisation techniques.

ANNs have a learning/training mechanism. According to this mechanism, the weights and biases for different connections of the neural network are changed so that the desired outputs are obtained. On the other hand, if the desired input and output samples are available by applying the input to the network and comparing the output of the network with the desired output, the weights and biases of the network are changed in such a way that the difference between the output of the network with the desired output reaches an acceptable level and so far, in previous papers, different stochastic and deterministic learning/training techniques have been introduced, e.g. gradient based [34] and PSO [35].

An example of the MLP is shown in Fig. 1. A transfer/system/network function is a mathematical representation and the relation between the input and output [36].

In this study, tangent (which has an output in the range of −1 to +1) and purelin (linear) functions were used in hidden and output layers, respectively, that are shown in Figs. 1 and 2. In addition, the output of MLP is calculated through the weights and biases according to three steps:

*Step 1*. Initially, the weighted totals of inputs are calculated as (4)

$$r_j = \sum_{i=1}^{l} (W_{ij}\,,\ Y_i) + B_j \tag{4}$$

where $l$ is the number of the input nodes, $W_{ij}$ is the connection weight from the $i$th input node to the $j$th hidden node, $h$ is the number of hidden nodes $(j = 1, 2, \ldots, h)$, $Y_i$ shows the $i$th input and $B_j$ is the bias of the $j$th hidden node.

*Step 2*. The output of each hidden node is calculated as (5)

$$R_j = \text{tansig}\,(r_j) = \frac{2}{1 + \exp(-2 \times r_j)} - 1 \tag{5}$$

*Step 3*. The final output is calculated according to (6) and (7)

$$f = \sum_{j=1}^{h} (W_j, R_i) + B' \tag{6}$$

$$F = \text{purelin}(f) = f \tag{7}$$

where $W_j$ is the connection weight from the $j$th hidden node to the output node and $B'$ is the bias of the output node. In this study, the ALO and PSO algorithms were utilised as a trainer for MLP's parameters.

## 5 Forecasting approach

The block diagram of a composite approach used in this study for MTLF shown in Fig. 3 and the following steps are taken:

*Step 1. Normalisation*. Usually, various data are used as inputs to the neural network. These data are normalised prior to applying to the neural network. Normalisation the data also increases the accuracy of prediction, prevents weight saturation in the training phase. The purpose of data normalisation is to have all the data between 0 and 1.

*Step 2. Modelling the data*. The purpose of data modelling is to construct the primary input matrix and target matrix. In fact, in order to apply the neural network and its training, the primary and secondary input matrices and also target matrix must be prepared. In this study depending on the total number of data available (***L*** and ***T*** matrices are shown in (8) and (9)), we choose the number of training samples (TS = 345) and the length of the back shift (BS = 15). It should be noted that the total number of TS and the length of the BS must be smaller or equal to the total number of data. The number of columns of the primary input matrix and the target matrix is considered equal to the number of TS. If only one type of data is used (e.g. load), the number of rows (the number of input variables) in the primary input matrix is equal to one length of the BS as (15) and if two different data are used (load and temperature), the number of rows in the primary input matrix is equal to two lengths of the BS as (16). The target matrix has only one row of predictive data and does not change with increasing data as (10). In this study, assuming the total number of data is 365 (daily peak load in one year) and the number of TS equal to 345 and the length of the BS=15, the structure of the primary input matrices and the target will be shown in (10) and (15)

$$\boldsymbol{L}_{1\times 365} = \begin{bmatrix} L_{t-364} & L_{t-363} & \cdots & L_{t-1} & L_t \end{bmatrix} \tag{8}$$

$$\boldsymbol{T}_{1\times 365} = \begin{bmatrix} T_{t-364} & T_{t-363} & \cdots & T_{t-1} & T_t \end{bmatrix} \quad (9)$$

$$\text{Target}_{1\times 345} = \begin{bmatrix} L_{t-344} & L_{t-343} & \cdots & L_{t-1} & L_t \end{bmatrix} \quad (10)$$

$$\rho_{X,Y} = COV(X,Y)/\left(\sigma_X \times \sigma_Y\right) \quad (11)$$

$$\text{COV}(X,Y) = E\left((X-\mu_x)\times\left(Y-\mu_y\right)\right) \quad (12)$$

$$\sigma_X = \sqrt{E(X-\mu_x)^2} \quad (13)$$

$$\sigma_Y = \sqrt{E\left(Y-\mu_y\right)^2} \quad (14)$$

*Step 3. Feature selection*. Preparing the secondary input matrix or decreasing the primary input matrix rows to $l$ from BS. Here, input variables that are similar to the target (effective input variables) and input variables that do not have the same effectiveness as other input variables (non-repetitive input variables) are selected. In this study, the correlation coefficient is used to measure the degree of dependence between the input variables in the primary input matrix and the target matrix. This method is used as a statistical tool to determine the type and degree of relationship with the other variable. The correlation coefficient between the two random variables $X$ and $Y$ is represented by $\rho_{X,Y}$ and calculated using (11). In (11)–(14), $\text{COV}(X,Y)$ is the covariance of $X$ and $Y$, in which $\mu_x$ and $\mu_y$ are expected values of variable $X$ and variable $Y$, respectively. In addition, $\sigma_X$ and $\sigma_Y$ are the standard deviation of variable $X$ and variable $Y$, respectively. Here, it should be noted that the limits infrared and radiofrequency (RF) are determined in such a way that the good data for input is not deleted. In summary, in the first step, the input variables are selected such that their resemblance to the target is greater than the threshold of irrelevancy filter (IF) and the output variables of the first filter are the input variables of the second filter (redundancy), and here, among the two variables similar to each other (over-threshold RF similarity), the variable is selected such that it is more similar to the target. Increasing the speed of ANN training and accuracy are from the advantages of feature selection

$$\text{Primary input}_{15\times 345} = \begin{bmatrix} L_{t-345} & L_{t-344} & \cdots & L_{t-2} & L_{t-1} \\ L_{t-346} & L_{t-345} & \ldots & L_{t-3} & L_{t-2} \\ \vdots & \ddots & \ldots & \vdots & \vdots \\ L_{t-359} & L_{t-358} & \cdots & L_{t-16} & L_{t-15} \end{bmatrix} \quad (15)$$

$$\text{Primary input}_{30\times 345} = \begin{bmatrix} T_{t-345} & T_{t-344} & \cdots & T_{t-2} & T_{t-1} \\ T_{t-346} & T_{t-345} & \cdots & T_{t-3} & T_{t-2} \\ \vdots & \ddots & \cdots & \vdots & \vdots \\ T_{t-359} & T_{t-358} & \cdots & T_{t-16} & T_{t-15} \\ L_{t-345} & L_{t-344} & \cdots & L_{t-2} & L_{t-1} \\ L_{t-346} & L_{t-345} & \cdots & L_{t-3} & L_{t-2} \\ \vdots & \ddots & \cdots & \vdots & \vdots \\ L_{t-359} & L_{t-358} & \cdots & L_{t-16} & L_{t-15} \end{bmatrix} \quad (16)$$

$$\text{or} \begin{bmatrix} L_{t-345} & L_{t-344} & \cdots & L_{t-2} & L_{t-1} \\ L_{t-346} & L_{t-345} & \cdots & L_{t-3} & L_{t-2} \\ \vdots & \ddots & \cdots & \vdots & \vdots \\ L_{t-359} & L_{t-358} & \cdots & L_{t-16} & L_{t-15} \\ T_{t-345} & T_{t-344} & \cdots & T_{t-2} & T_{t-1} \\ T_{t-346} & T_{t-345} & \cdots & T_{t-3} & T_{t-2} \\ \vdots & \ddots & \cdots & \vdots & \vdots \\ T_{t-359} & T_{t-358} & \cdots & T_{t-16} & T_{t-15} \end{bmatrix}$$

*Step 4. Shuffling*. The shuffling of the TS considering that the TS in the secondary input matrix and target are in order of time to prevent the training process from slowing down and reduce the accuracy.

*Step 5. Training and testing*. Training or learning is the construction of a mapping between the secondary input matrix with $l$ rows and the target and testing this template for real data. A flowchart of optimal training MLP is shown in Fig. 4.

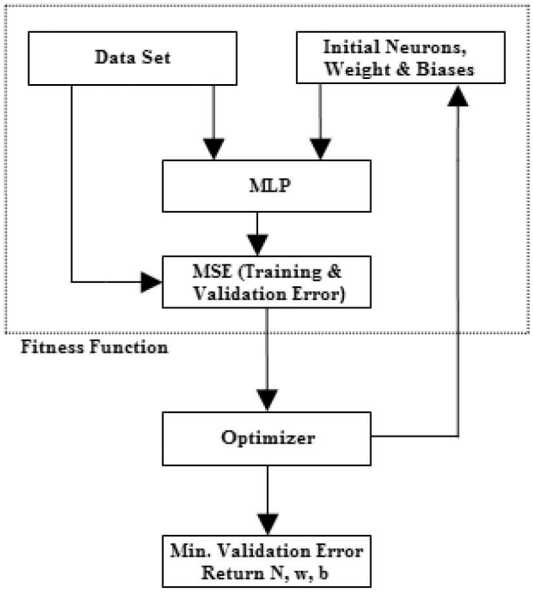


**Fig. 4** *Flowchart of optimal training MLP*

In this study, a multi-layer neural (MLP) with the specifications given in the previous section was used to obtain this template and the ALO and PSO algorithms were used to determine the number of hidden neurons ($N$), the limit of thresholds (IF, RF) and the weights and biases ($w$, $b$) so that the training error (mean square error (MSE)) and the testing error(mean absolute percentage error (MAPE)) are minimised. In other words, the purpose of using these optimisers is to find the best answer for these variables, so that they provide the least error rate at both training and testing samples subject to some constraints ($10 \le N \le 30$, $0.25 \le \text{IF} \le 0.55$, $0.6 \le \text{RF} \le 0.95$, $0 \le w, b \le 1$).

MSE and MAPE are used to measure the deviation of the actual value of the forecasted value, shown in (17) and (18). $f_j^i$ and $o_j^i$ are the forecasted and actual values, respectively, where the $j$th input variable and the $i$th training or testing sample are used.

$$\text{MSE} = \frac{1}{\text{TS}}\sum_{i=1}^{\text{TS}}\left(o_j^i - f_j^i\right)^2 \quad (17)$$

$$\text{MAPE} = \frac{1}{\text{TS}}\sum_{i=1}^{\text{TS}}\frac{\left|o_j^i - f_j^i\right|}{o_j^i} \times 100 \quad (18)$$

Daily peak load has a lot of changes that can be seen from Fig. 5. Owing to the complexity of daily peak load and the limited number of TS in MTLF, MLPs typically require a large number of hidden weights and biases to prevent the divergence of the optimisation algorithm and saturation of the training capability. On the other hand, the number of training iteration has a special sensitivity. If the number of training iteration is low, training is incomplete, and weights and biases are not properly corrected and if the number of training iteration is high, the MLP memorises the relationship (map) between the input matrix and the target instead of learning it. To solve the problem, a cross-validation method has been used [2]. In this method, first, a part of the training sample is separated and considered as the validation set. With the number of iterations, training error and validation error are reduced, but validation error starts to increase from one point (minimum validation error). This is the best point to show optimal training iterations. In other words, the minimum error of the validation set is selected as the optimal point in the training.

*Step 6. Analysis of results*.

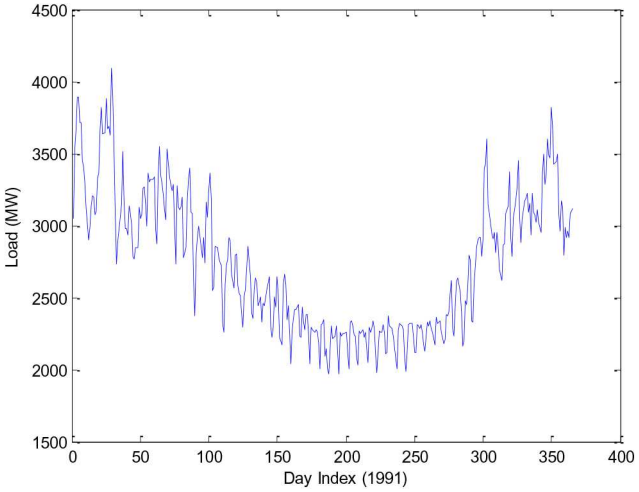


**Fig. 5** *Daily peak load profile of North-American electric utility*

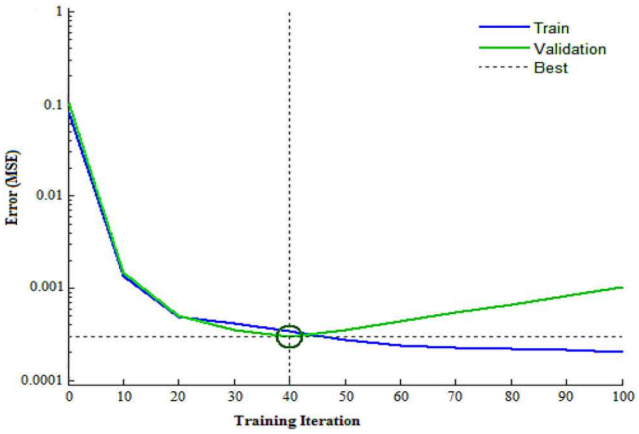


**Fig. 6** *Training error and validation error*

## 6 Simulation results and discussions

In this study, the daily peak load from two datasets is used to prove the effectiveness of the combined method. In the first dataset, the hourly load series ranging from 1 January 1987 to 31 December 1991 is available belonging to a North-American electric utility [10]. To do this study, the daily peak load is calculated from 1987 to 1991. In the second dataset, daily peak local load in Kerman, Iran from 2016 to 2017 is available.

### *6.1 North-American electric utility*

For the construction of MTLF with monthly forecast horizon and daily forecast step, the data required to implement all steps presented in Section 5, related to the hourly load of 1987–1991 by a North American electricity utility. Then the daily peak times are calculated for these hourly loads.

In the first case, the 1990 peak daily data for neural network training was used and in the second case, the data of the same seasons in the past four years (1987–1990) was applied as the training set to establish the forecasting model (seasonal training). Specifically, we have chosen four months including two months before the target month and one month after the target month as the same season, to ensure that the collected data is sufficient for training purposes and the seasonal patterns are captured. To verify the performance of our forecasting approach, four months in 1991 (i.e. January, April, July, and October), which represent summer, winter, and transitional periods of the year, were selected as the testing periods. In this study, in the algorithms, $t_{max} = 100$, $N_p = 30$ and the parameters of the PSO algorithm: $c_1$, $c_2$, $\omega_{max}$ and $\omega_{min}$ are set to 2, 2, 0.9, and 0.4, respectively.

The simulation of the proposed method is performed with MATLAB 8.1.0 (R2013a) and after 20 times the run of PSO and ALO algorithms, the average and minimum of the obtained MAPE are reported.

The training and validation errors of a training phase of the MLP for the case study are shown in Fig. 6 and in Figs. 7–9 the real data is compared with forecasted data in a month (daily peak) for different optimisers based on one year/seasonal training.

Comparison of MAPE and MSE by different optimisers (ALO/PSO) and different functions, e.g. Levenberg–Marquardt (LM), Bayesian regulation (BR), Broyden-Fletcher-Goldfarb-Shanno (BFGS) quasi-Newton (known as Trainbfg train function) based on the historical daily peak load data (one year) and the peak daily load (seasonal) for MLP training for four months (were selected as the testing periods) have been shown in Tables 1 and 2, respectively. Also, these results and the best result of [10] in this regard, but in different methods, are presented in Fig. 10. The selected inputs feature or the number of rows in the secondary input matrix to apply the neural network by the different optimisers in a month is reported in Table 3.

Simulation results reveal that

i. If the results of this study are compared with the best results in another article and from different methods [10] (Fig. 10), we found that the proposed method used in this study gives better and more acceptable results.
ii. ALO and PSO algorithms create promising results but the results of ALO are better than PSO.
iii. Among the studied methods of stochastic training such as seasonal/one year training with the ALO or PSO algorithms, the seasonal training model achieved the lowest MSE and MAPE in the investigated case study.
iv. Significant improvement of the ALO/PSO trainer over the other trainers is the result of their high exploration and exploitation.

### 6.2 Local load in Kerman, Iran

After applying the proposed method on data from a North American utility and comparing the results with another reference [10], it was shown that the proposed hybrid method is acceptable. So, for further testing, also implement the problem for local load in Kerman. Fig. 11 indicates the daily peak local load profile of 2017. Owing to having only two years of data from 2016 to 2017, only one year of training was performed. Really in this case, the 2016 peak daily data for neural network training was used. To verify the performance of our forecasting approach, four months in 2017 (i.e. January, April, July, and October) were selected as the testing periods and the results are presented in Table 4. The results show that the proposed method for a local load in Kerman is also acceptable.

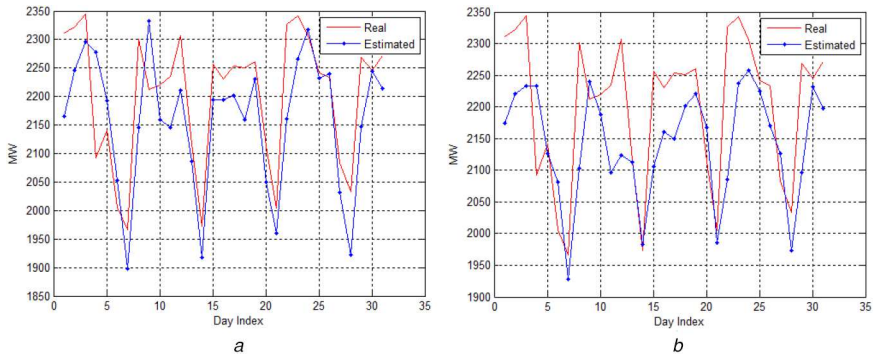


**Fig. 7** *Comparison of real and forecasted data in July 1991 for MLP(LM)*
***(a)*** Seasonal training, ***(b)*** One year training

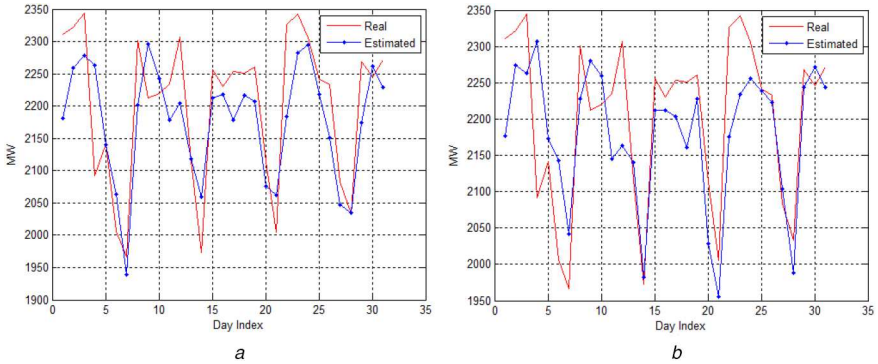


**Fig. 8** *Comparison of real and forecasted data in July 1991 for MLP(PSO)*
***(a)*** Seasonal training, ***(b)*** One year training

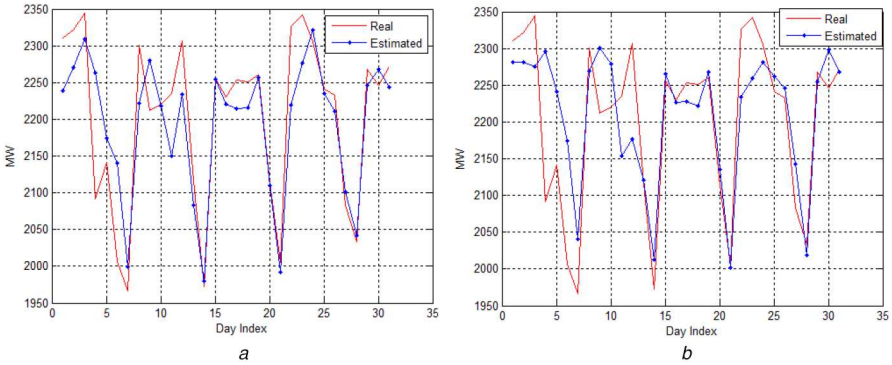


**Fig. 9** *Comparison of real and forecasted data in July for MLP(ALO)*
***(a)*** Seasonal training, ***(b)*** One year training

**Table 1** Comparison of MAPE and MSE based on the daily peak data (one year training)

| Months | MSE | | | MAPE (%) | | | | |
|---|---|---|---|---|---|---|---|---|
| | ALO | PSO | Index | MLP(LM) | MLP(BR) | MLP(BFG) | MLP(ALO) | MLP(PSO) |
| January | 0.000792 | 0.000803 | min. | 5.13 | 5.11 | 5.17 | 4.69 | 5.1 |
| | | | mean | 11.79 | 5.79 | 9.63 | 4.82 | 5.64 |
| April | 0.000314 | 0.000434 | min. | 4.63 | 4.42 | 4.36 | 4.29 | 3.94 |
| | | | mean | 6.53 | 4.81 | 5.75 | 4.76 | 4.58 |
| July | 0.000688 | 0.000491 | min. | 3.68 | 2.77 | 2.94 | 2.37 | 2.91 |
| | | | mean | 4.22 | 3.16 | 3.75 | 3.11 | 3.43 |
| October | 0.000363 | 0.000492 | min. | 5.49 | 5.78 | 5.6 | 4.3 | 5.22 |
| | | | mean | 6.16 | 6.04 | 6.68 | 5.03 | 5.67 |

**Table 2** Comparison of MAPE and MSE based on historical daily peak load data (seasonal training)

| Months | MSE | | | MAPE (%) | | | | |
|---|---|---|---|---|---|---|---|---|
| | ALO | PSO | Index | MLP(LM) | MLP(BR) | MLP(BFG) | MLP(ALO) | MLP(PSO) |
| January | 0.000725 | 0.000784 | min. | 4.56 | 4.48 | 4.49 | 3.73 | 3.88 |
| | | | mean | 5.74 | 5.01 | 6.15 | 3.88 | 4.18 |
| April | 0.000351 | 0.000228 | min. | 3.84 | 3.66 | 3.86 | 3.03 | 3.37 |
| | | | mean | 4.71 | 4.16 | 5.11 | 3.46 | 3.48 |
| July | 0.000683 | 0.000486 | min. | 1.9 | 2.27 | 2.35 | 1.87 | 1.88 |
| | | | mean | 2.76 | 3.26 | 3.27 | 1.97 | 2.03 |
| October | 0.000699 | 0.000126 | min. | 4.89 | 5.59 | 4.62 | 3.55 | 3.59 |
| | | | mean | 6.42 | 7.73 | 6.27 | 3.94 | 4.09 |

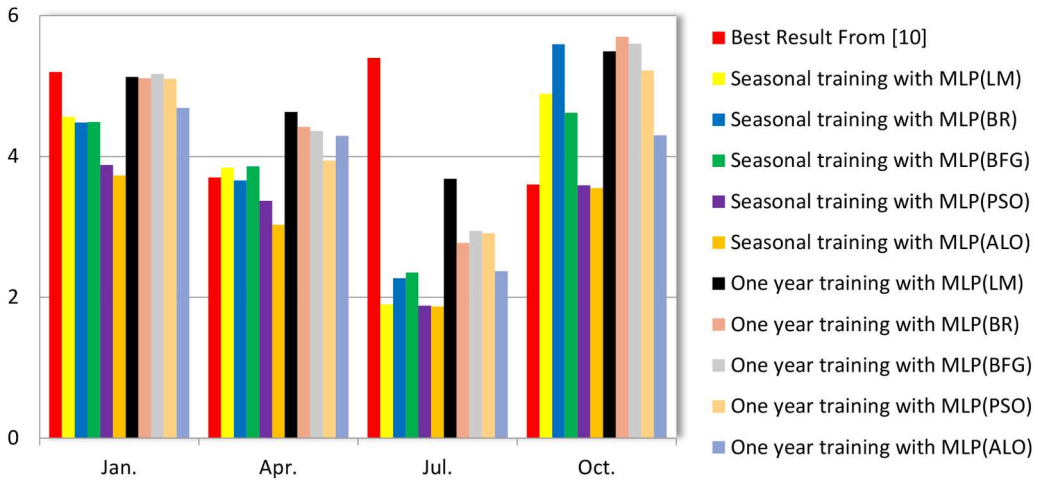


**Fig. 10** *Comparison of MAPE (%) by different optimiser/function based on one year/seasonal peak daily training (four months were selected as the testing periods)*

**Table 3** Selected input features with the proposed method in a month (July 1991)

| | Filter | MLP(ALO) | MLP(PSO) |
|---|---|---|---|
| one year training | irrelevancy | 1, 2, 6, 7, 14 | 1, 2, 3, 6, 7, 8, 14 |
| | redundancy | 1, 7, 14 ($I = 3$) | 1, 3, 7, 14 ($I = 4$) |
| seasonal training | irrelevancy | 1, 2, 6, 7, 8, 14 | 1, 2, 3, 5, 6, 7, 8, 14 |
| | redundancy | 1, 7 ($I = 2$) | 1, 3, 5, 6, 7 ($I = 5$) |

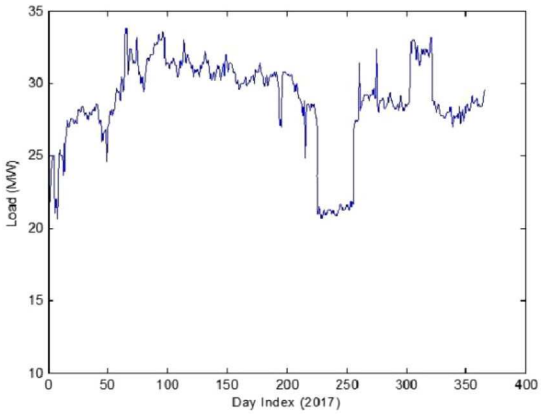


**Fig. 11** *Daily peak load profile of 2017 (local load in Kerman)*

**Table 4** Comparison of MAPE based on the 2016 peak daily data from the local load (one year training)

| Four months in 2017 | MAPE (%) | | | | |
|---|---|---|---|---|---|
| | MLP(LM) | MLP(BR) | MLP(BFG) | MLP(ALO) | MLP(PSO) |
| January | 5.6459 | 4.3493 | 5.9435 | 3.4563 | 4.0680 |
| April | 3.1046 | 2.1958 | 3.6410 | 1.3793 | 1.7681 |
| July | 3.0292 | 3.4585 | 3.2723 | 2.3780 | 2.3679 |
| October | 2.0440 | 2.3389 | 2.2801 | 1.4959 | 1.5487 |

## 7 Conclusions

One of the important goals of MTLF is planning in power systems. On the other hand, scheduling of peak times is very important in the operation and reliability of the system. Therefore, the monthly forecast horizon and daily forecast step (daily peak load) were used for MTLF.

In this study, the ALO/PSO was proposed as a stochastic trainer to the MLP. The problem of training the MLP was formulated for the ALO/PSO algorithm to find the optimal values for the weights and biases. Other targets in this study are to determine the optimal data to apply the neural network as well as to create the optimal structure in the neural network so that the validation error is minimised. By optimising the input data while increasing the forecast accuracy, it also reduces the computational time of the problem. Creating the optimal structure of the neural network is actually the determination of the number of neurons in the middle layer which is considered as a decision variable in the ALO and PSO optimisation algorithms.

According to the results, ALO and PSO algorithms produce promising results due to the high exploration and exploitation of the ALO/PSO but the results of ALO are better than PSO and among the studied methods of stochastic training such as seasonal/one year training with the ALO or PSO algorithms, the seasonal training model achieved the lowest MSE and MAPE in two investigated case studies. Finally, the results of this study from the first dataset were compared with the best results in another article and showed that the proposed method was used in this study to give better and more acceptable results. In addition, local daily peak load was used to more test and prove the effectiveness of the combined method.

In this study, the daily peak load of a month from the next year forecasted due to the complexity of the daily peak load characteristic and found that the proposed method has high accuracy. Therefore, according to the complexity of the price characteristics, this method can also be used to forecast the mid-term price. Therefore, the proposed method can select the optimal set of input variables based on a new optimisation algorithm. In addition, the proposed method can obtain the optimal structure for neural networks as a basic predictor by using a new optimisation method. Specially, the proposed method can train the load predictor weights optimally based on a new optimisation algorithm.

The obtained results show the proposed method ability and accuracy outperforms other methods.

## 8 References


[1] Khuntia, S.R., Rueda, J.L., vander Meijden, M.A.M.M.: 'Forecasting the load of electrical power systems in mid and long-term horizons – a review', *IET Gener. Transm. Distrib.*, 2016, **10**, pp. 3971–3977

[2] Amjady, N., Keynia, F.: 'Mid-term load forecasting of power systems by a new prediction method', *Energy Convers. Manage.*, 2008, **49**, (10), pp. 2678–2687

[3] Hippert, H.S., Pedreira, C.E., Souza, R.C.: 'Neural networks for short-term load forecasting: a review and evaluation', *IEEE Trans. Power Syst.*, 2001, **16**, (1), pp. 44–55

[4] Lee, W.J., Hong, J.: 'A hybrid dynamic and fuzzy time series model for mid-term power load forecasting', *Electr. Power Energy Syst.*, 2015, **64**, pp. 1057–1062

[5] Smola, A.J., Schölkopf, B.: 'A tutorial on support vector regression', *Stat. Comput.*, 2004, **14**, (3), pp. 199–222

[6] Hong, W.C., Dong, Y., Lai, C.Y., *et al.*: 'SVR with hybrid chaotic immune algorithm for seasonal load demand forecasting', *Energies*, 2011, **4**, (6), pp. 960–977

[7] Zhang, Z., Ye, S.: 'Long term load forecasting and recommendations for China based on support vector regression'. Proc. IEEE Information Management, Innovation Management and Industrial Engineering, Shenzhen, People's Republic of China, 26–27 November 2011

[8] Wang, J., Li, L., Niu, D., *et al.*: 'An annual load forecasting model based on support vector regression with differential evolution algorithm', *Appl. Energy*, 2012, **94**, pp. 65–70

[9] Hong, W.C., Dong, Y., Zhang, W.Y., *et al.*: 'Cyclic electric load forecasting by seasonal SVR with chaotic genetic algorithm', *Electr. Power Energy Syst.*, 2013, **44**, (1), pp. 604–614

[10] Hu, Z., Bao, Y., Chiong, R., *et al.*: 'Mid-term interval load forecasting using multi-output support vector regression with a memetic algorithm for feature selection', *Energy*, 2015, **84**, pp. 419–431

[11] Ghiassi, M.D., Zimbra, D.K., Saidane, H.: 'Medium term system load forecasting with a dynamic artificial neural network model', *Electr. Power Syst. Res.*, 2006, **76**, (5), pp. 302–316

[12] Elkateb, M.M., Solaiman, K., Al-Turki, Y.: 'A comparative study of medium-weather-dependent load forecasting using enhanced artificial/fuzzy neural network and statistical techniques', *Neurocomputing*, 1998, **23**, (1), pp. 3–13

[13] Ghofrani, M., Ghayekhloo, M., Arabali, A., *et al.*: 'A hybrid short term load forecasting with new input selection framework', *Energy*, 2015, **81**, pp. 777–786

[14] Hahn, H., Meyer-Nieberg, S., Pickl, S.: 'Electric load forecasting methods: tools for decision making', *Eur. J. Oper. Res.*, 2009, **199**, pp. 902–907

[15] Fajardo-Toro, C.H., *et al.*: 'Adaptive and hybrid forecasting models – a review', in '*Engineering digital transformation*' (Springer, 2018), pp. 315–322

[16] De Aquino, R.R., Neto, O.N., Lira, M., *et al.*: 'Development of an artificial neural network by genetic algorithm to mid-term load forecasting'. Proc. IEEE Int. Joint Conf. on Neural Networks, Orlando, FL, USA, August 12–17, 2007

[17] Karabulut, K., Alkan, A., Yilmaz, A.S.: 'Long term energy consumption forecasting using genetic programming', *Math. Comput. Appl.*, 2008, **13**, (2), pp. 71–80

[18] Li, H.Z., Guo, S., Li, C.J., *et al.*: 'A hybrid annual power load forecasting model based on generalized regression neural network with fruit fly optimization algorithm', *Knowl.-Based Syst.*, 2013, **37**, pp. 378–387

[19] AlRashidi, M.R., El-Naggar, K.M.: 'Long term electric load forecasting based on particle swarm optimization', *Appl. Energy*, 2010, **87**, (1), pp. 320–326

[20] Zebarjadi, M., Askarzadeh, A.: 'Optimization of a reliable grid connected PV-based power plant with/without energy storage system by a heuristic approach', *Sol. Energy*, 2016, **125**, pp. 12–21

[21] Chen, M.Y., Chen, B.T.: 'A hybrid fuzzy time series model based on granular computing for stock price forecasting', *Inf. Sci.*, 2015, **294**, pp. 227–241

[22] Jiang, P., Liu, F., Song, Y.L.: 'A hybrid forecasting model based on date-framework strategy and improved feature selection technology for short-term load forecasting', *Energy*, 2017, **119**, pp. 694–709

[23] Chahkoutahi, F., Khashei, M.: 'A seasonal direct optimal hybrid model of computational intelligence and soft computing techniques for electricity load forecasting', *Energy*, 2017, **140**, pp. 988–1004

[24] Chen, K., Chen, K., Wang, Q., *et al.*: 'Short-term load forecasting with deep residual networks', *IEEE Trans. Smart Grid*, 2018, **10**, (4), pp. 3943–3952

[25] Eberhart, R., Kennedy, J.: 'A new optimizer using particle swarm theory'. Proc. Sixth Int. Symp. on Micro Machine and Human Science, Nagoya, Japan, 1995, pp. 39–43

[26] Eberhart, R., Shi, Y.: 'Empirical study of particle swarm optimization'. Proc. Congress on Evolutionary Computation (CEC99), Washington, DC, USA, 1999, pp. 1945–1950

[27] Mirjalili, S.: 'The ant lion optimizer', *Adv. Eng. Softw.*, 2015, **83**, pp. 80–98

[28] McCulloch, W.S., Pitts, W.: 'A logical calculus of the ideas immanent in nervous activity', *Bull. Math. Biophys.*, 1943, **5**, (4), pp. 115–133

[29] Bebis, G., Georgiopoulos, M.: 'Feed-forward neural networks', *IEEE Potentials*, 1994, **13**, (4), pp. 27–31

[30] Dorffner, G.: 'Neural networks for time series processing', *Neural Netw. World*, 1996, **6**, pp. 447–468

[31] Ghosh-Dastidar, S., Adeli, H.: 'Spiking neural networks', *Int. J. Neural Syst.*, 2009, **19**, (4), pp. 295–308

[32] Kohonen, T.: 'The self-organizing map', *Neurocomputing*, 1998, **21**, (1), pp. 1–6

[33] Park, J., Sandberg, I.W.: 'Approximation and radial-basis-function networks', *Neural Comput.*, 1993, **5**, (2), pp. 305–316

[34] Huang, G.-B., Zhu, Q.-Y., Siew, C.-K.: 'Extreme learning machine: a new learning scheme of feed forward neural networks'. Proc. IEEE Int. Joint Conf. on Neural Networks, Budapest, Hungary, 25–29 July 2004, vol. 2, pp. 985–990

[35] Mendes, R., Cortez, P., Rocha, M., *et al.*: 'Particle swarms for feed forward neural network training', Proc. Int. Joint Conf. Neural Netw., Honolulu, HI, USA, May 2002, vol. 13, pp. 1895–1899

[36] Yitian, L., Gu, R.R.: 'Modeling flow and sediment transport in a river system using an artificial neural network', *Environ. Manage.*, 2003, **31**, (1), pp. 122–134