Training Feedforward Neural Networks Using Social Learning Particle Swarm Optimization-A Case Comparison Study on Electrical System

Yuansheng Huang¹, Ying Qiao¹²*, Yuelin Gao²

ABSTRACT
The knowledge about training feed forward neural networks (FNNs) is an important and complex issue in the supervised learning field. In the process of learning, the FNNs system involves some input parameters such as connection weights and biases, which may greatly influence the performance of FNNs training. In this paper, a newly developed meta-heuristic method, named social learning particle swarm optimization (SLPSO), is trying to find the optimal combination of connection weights and biases for FNNs, which is often used to deal with power load forecasting problem. In the numerical experiments, a case on the power load forecasting problem is employed to verify the effectiveness of SLPSO. The experiment results indicate that SLPSO has the advantages on the training accuracy and testing accuracy with respect to other six state-of-the-art intelligent optimization algorithms.

Key Words: Training Feed Forward Neural Networks, Social Learning Particle Swarm Optimization, Particle Swarm Optimization, Electrical System

DOI Number: 10.14704/nq.2018.16.6.1588

Introduction
In recent years, artificial neural networks (ANNs), as a kind of effective computing tools, has been researched extensively in the scientific community because of its powerful generalization ability, learning capability and adaptability. By now, it has been successfully applied to many areas such as associative memories, pattern matching, function approximation, pattern classification and so on (Dayhoff et al., 1990; Ghritlahre and Prasad, 2018; Isah et al., 2017). As one of the most popular ANNs, feed forward neural networks (FNNs), especially FFNs with three layers, has been proven that it is suitable for classification problems of nonlinearly separable patterns (Lin et al., 2004; Isa et al., 2011). However, the classification performance of FNNs depends heavily on the learning process, which is closely related to the selected connection weights and biases. As a result, the topic about how to find the optimal combination of connection weights and biases for FNNs attracts more and more attention. Many times, some traditional optimization techniques such as the back-propagation algorithm (Rumelhart et al., 1986) which is a gradient-based method to minimize the learning error, are suggested to solve the problem of FNNs training. However, these gradient-based algorithms expose some limitations: the first is that the training results depend greatly on the initial weights and the second is that they are easy to fall into local optimum that is not global (Montana et al., 1989).

Compared with gradient-type methods, nature-inspired stochastic algorithms have the advantages on derivation free, and the high

Corresponding author: Ying Qiao
Address: ¹ Economics and Management, North China Electric Power University, Beijing 102206, China; ² Mathematics and Information Science, Beifang University of Nationalities, Ningxia 750021, China
e-mail xjqiao88@163.com

Relevant conflicts of interest/financial disclosures: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.
Received: 3 March 2018; Accepted: 27 April 2018
A brief introduction of FNNs

In this paper, the FNNs with three layers (including one input layer, one hidden layer, and one output layer) is considered for training power load forecasting. Its network structure is shown in Figure 1. It can be seen from the Figure 1 that the information of FNNs diffuses from an input layer to an output layer, which can be achieved by hidden layers with connection weights and activation functions. To be specific, the output of each hidden unit in each epoch of learning is carried on by sigmoid activation function. The calculation formula of sigmoid activation function is written below:

$$f(x) = \frac{1}{1 + \exp\left(-\sum \omega_j x_j - \theta_j\right)}$$

(1)

where $x(i=1, 2, ..., n)$ is the $i$th input variable, $n$ is the number of input units. $\omega_j$ is the corresponding connection weight from the $i$th input unit to the $j$th hidden unit, $\theta_j$ is the bias of the $j$th hidden unit. $s = \omega_jx_j - \theta_j(i=1, 2, ..., h)$ is the $j$th hidden unit, $h$ is the number of hidden units.

Figure 1. The network structure of FNNs with three layers

After the calculation outputs of hidden units, the last results of output layer can be calculated as follows:

$$o_k = \sum_{j=1}^{h} \tilde{\omega}_{jk} f(s_j) - \tilde{\theta}_k$$

(2)

where $o_k(k=1, 2, ..., m)$ is the $k$th output result, $m$ is the number of output units. $\tilde{\omega}_{jk}$ is the corresponding connection weight from the $j$th hidden unit to the $k$th output unit, and $\tilde{\theta}_k$ is the bias of the $k$th output unit.

Without loss of generality, the learning error MSE of FNNs training can be expressed as follows:

$$MSE = \frac{1}{M} \sum_{i=1}^{M} (\sum_{k=1}^{m} (o_{ik} - d_{ik})^2)$$

(3)

where $M$ is the total number of training sampling, and $D_{ik}$ is the desired output of the $i$th input unit.
As a result, FNNs training problem can be formalized below:

\[
\begin{align*}
\text{min} \quad & \sum_{a=1}^{M} \sum_{i=1}^{n_h} (y_a - d_a)^2 \\
\text{s.t.} \quad & \omega_l^i \leq \omega_i^j \leq \omega_u^i, \ i = 1, 2, \ldots, nh \\
& \theta_l^j \leq \theta_i^j \leq \theta_u^j, \ i = 1, 2, \ldots, h \\
& \theta_l^q \leq \theta_i^q \leq \theta_u^q, \ q = 1, 2, \ldots, m
\end{align*}
\] (4)

Swarm sorting and behavior learning

From the Figure 2, it can be seen that behavior learning is the most important component in SLPSO apart from the fitness evaluations and swarm sorting. In order to make an easy description on the behavior learning mechanisms, the swarm is first sorted based on an ascending order of the particles’ fitness values. Each particle or imitator (except the best one) will then learn from its corresponding demonstrators. It is noted that a particle will be serve a demonstrator for different imitators more than once in each generation. For example, for the \( i \)th imitator, its demonstrators can be selected from any particle \( k \) that satisfies \( i,k \leq N \).

Inspired from social learning mechanism (R. Cheng et al., 2015), an imitator will learn the behaviors from different demonstrators as follows:

\[
x'_i(t+1) = \begin{cases} 
    x'_i(t) + \Delta x'_i(t+1), \text{if } rand \leq P_i^l \\
    x'_i(t), \text{otherwise}
\end{cases}
\] (5)

where \( x'_i(t)(i=1, 2, \ldots, N; j=1, 2, \ldots, D) \) is the \( j \)th dimension of \( i \)th behavior vector at the \( t \)th generation, \( D \) is the problem dimension, \( P_i^l \) is the learning probability of \( i \)th particle, \( rand \) is a random number between 0 and 1, and \( \Delta x'_i(t+1) \) is the behavior correction which can be generated below:

\[
\Delta x'_i(t+1) = rand_{1}(t) \cdot \Delta x'_i(t) + rand_{2}(t) \cdot I_1(t) + rand_{3}(t) \in C_1(t)
\] (6)

with

\[
\begin{align*}
I_1(t) &= x'_i(t) - x'_j(t); \\
C_1(t) &= \bar{x}_j(t) - x'_j(t).
\end{align*}
\] (7)

From the Eq. (6), it can be seen that the behavior correction \( \Delta x'_i(t+1) \) is made up of three components. The first component \( \Delta x'_i(t) \) called the inertia component is the same as the canonical PSO. The second component \( I_1(t) \) is denoted as imitation component which replaces the learning from pbest as done in the canonical PSO. The third component \( C_1(t) \) named social influence component is used to make the mean behavior of all particles in the current population to replace the learning from gbest as done in the canonical PSO, in which \( \bar{x}_j(t) = \frac{1}{N} \sum_{i=1}^{N} x'_i(t) \). rand\(_1\)(t), rand\(_2\)(t), rand\(_3\)(t)
\(\text{rand}(t)\) are three random coefficients generated within \([0, 1]\), and \(c\) is the social influence factor.

**Figure 3.** Main components of SLPSO

**Methods**

**A case study on power load forecasting problem**

Load forecasting is very significant and useful in the construction and operation of power systems. Prediction accuracy directly affects the safety operation of power systems and economic benefits. In this paper, the electricity consumption \(y\) from 1990 to 2008 in some province of China is selected as the experimental data (Zhuang et al., 2012). Its related factors are gross domestic product (GDP)-\(x_1\), per capital (GDP)-\(x_2\), GDP growth-\(x_3\), the total industrial value-\(x_4\), gross fixed asset formation-\(x_5\), agricultural electricity-\(x_6\), the total value of agriculture and forestry and animal husbandry fish-\(x_7\). The history data from 1990 to 2008 is listed in Table 1. In this case, the history data from 1990 to 2000 is selected as training set and the rest is the test set. In order to eliminate the adverse effect by power negative impact factors with different dimension, these data will be normalized by the following formula:

\[
\hat{x}_i = \frac{x_i - x^\text{min}_i}{x^\text{max}_i - x^\text{min}_i}
\]

where \(x_i (i=1, 2, ..., 7)\) is the original value of history data, \(\hat{x}_i\) is the value of \(x_i\) normalized, \(x^\text{max}_i\) and \(x^\text{min}_i\) are maximum value and minimum value, respective.

**Experimental platform and parameter setting**

For all experiments, 25 independent runs are carried out on the same machine with a Celoron 3.40 GHz CPU, 4.00 GB memory, and windows 7 operating system with Matlab 7.9, and conducted with the maximum number of function evaluations (\(\text{MAX\_FES}\)) as the termination criterion. The goal is to ensure a fair comparison and reduce the statistical error. In the numerical experiments, \(\text{MAX\_FES}\) is set to 50,000 for all competition algorithms. The parameters of these algorithms agree well with the original papers.

**Performance metric**

In our experimental studies, the mean value and standard deviation (SD) of training error and test error are recorded for evaluating the performance of each algorithm.

**Table 1.** The power load data from 1990 to 2008

<table>
<thead>
<tr>
<th>Year</th>
<th>(x_1)</th>
<th>(x_2)</th>
<th>(x_3)</th>
<th>(x_4)</th>
<th>(x_5)</th>
<th>(x_6)</th>
<th>(x_7)</th>
<th>(y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1990</td>
<td>423.2</td>
<td>2116.3</td>
<td>24.1</td>
<td>411.7</td>
<td>670266.4</td>
<td>18.0</td>
<td>202.3</td>
<td>57.07</td>
</tr>
<tr>
<td>1991</td>
<td>597.1</td>
<td>2661.3</td>
<td>32.5</td>
<td>504.4</td>
<td>794318.4</td>
<td>17.8</td>
<td>248.7</td>
<td>584.3</td>
</tr>
<tr>
<td>1992</td>
<td>760.9</td>
<td>3368.0</td>
<td>34.4</td>
<td>713.6</td>
<td>893213.5</td>
<td>22.5</td>
<td>346.3</td>
<td>607.6</td>
</tr>
<tr>
<td>1993</td>
<td>930.1</td>
<td>4069.5</td>
<td>28.1</td>
<td>874.0</td>
<td>921501.3</td>
<td>21.1</td>
<td>421.3</td>
<td>643.4</td>
</tr>
<tr>
<td>1994</td>
<td>1100.9</td>
<td>4763.6</td>
<td>22.1</td>
<td>1047.4</td>
<td>977180.2</td>
<td>23.4</td>
<td>519.5</td>
<td>671.4</td>
</tr>
<tr>
<td>1995</td>
<td>1221.2</td>
<td>5234.1</td>
<td>14.8</td>
<td>1201.9</td>
<td>1074699.9</td>
<td>21.6</td>
<td>547.7</td>
<td>700.0</td>
</tr>
<tr>
<td>1996</td>
<td>1333.7</td>
<td>5670.9</td>
<td>13.3</td>
<td>1269.2</td>
<td>1134627.4</td>
<td>23.3</td>
<td>600.1</td>
<td>729.5</td>
</tr>
<tr>
<td>1997</td>
<td>1416.5</td>
<td>5966.5</td>
<td>11.7</td>
<td>1557.4</td>
<td>1303136.7</td>
<td>24.6</td>
<td>595.6</td>
<td>757.8</td>
</tr>
<tr>
<td>1998</td>
<td>1562.8</td>
<td>6550.6</td>
<td>15.1</td>
<td>1271.1</td>
<td>1524550.7</td>
<td>26.2</td>
<td>607.4</td>
<td>808.8</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>2004</td>
<td>1959.8</td>
<td>8124.6</td>
<td>81.5</td>
<td>2329.0</td>
<td>2453267.8</td>
<td>56.0</td>
<td>687.2</td>
<td>890.1</td>
</tr>
<tr>
<td>2005</td>
<td>1973.5</td>
<td>8137.3</td>
<td>90.5</td>
<td>2345.0</td>
<td>2453267.1</td>
<td>69.5</td>
<td>694.6</td>
<td>895.9</td>
</tr>
<tr>
<td>2006</td>
<td>1971.5</td>
<td>8139.9</td>
<td>97.8</td>
<td>2350.3</td>
<td>2453281.9</td>
<td>74.9</td>
<td>700.9</td>
<td>909.2</td>
</tr>
<tr>
<td>2007</td>
<td>1976.2</td>
<td>8151.4</td>
<td>105.7</td>
<td>2360.1</td>
<td>2453291.3</td>
<td>92.5</td>
<td>707.6</td>
<td>915.7</td>
</tr>
<tr>
<td>2008</td>
<td>1978.9</td>
<td>8162.7</td>
<td>114.8</td>
<td>2367.1</td>
<td>2453293.4</td>
<td>108.9</td>
<td>716.2</td>
<td>933.7</td>
</tr>
</tbody>
</table>
Table 2. The training error of eight POS-BASED algorithms over 25 independent runs on power load forecasting problem with 50,000 FES.

<table>
<thead>
<tr>
<th>Algorithm/Result</th>
<th>min</th>
<th>max</th>
<th>median</th>
<th>mean</th>
<th>std</th>
<th>p</th>
<th>h</th>
<th>significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSO</td>
<td>1.70E-02</td>
<td>2.87E-02</td>
<td>2.19E-02</td>
<td>2.23E-02</td>
<td>2.40E-03</td>
<td>1.42E-09</td>
<td>1</td>
<td>‡</td>
</tr>
<tr>
<td>BBPSO</td>
<td>2.03E-02</td>
<td>5.14E-02</td>
<td>2.79E-02</td>
<td>3.15E-02</td>
<td>8.96E-03</td>
<td>1.42E-09</td>
<td>1</td>
<td>‡</td>
</tr>
<tr>
<td>CLPSO</td>
<td>3.01E-02</td>
<td>6.32E-02</td>
<td>4.03E-02</td>
<td>4.21E-02</td>
<td>9.41E-03</td>
<td>1.42E-09</td>
<td>1</td>
<td>‡</td>
</tr>
<tr>
<td>QPSO</td>
<td>1.34E-02</td>
<td>2.32E-02</td>
<td>1.85E-02</td>
<td>1.83E-02</td>
<td>2.48E-03</td>
<td>1.42E-09</td>
<td>1</td>
<td>‡</td>
</tr>
<tr>
<td>MRPSO</td>
<td>1.50E-02</td>
<td>2.91E-02</td>
<td>1.98E-02</td>
<td>1.98E-02</td>
<td>3.06E-03</td>
<td>1.42E-09</td>
<td>1</td>
<td>‡</td>
</tr>
<tr>
<td>UPSO</td>
<td>2.06E-02</td>
<td>3.30E-02</td>
<td>2.79E-02</td>
<td>2.74E-02</td>
<td>3.55E-03</td>
<td>1.42E-09</td>
<td>1</td>
<td>‡</td>
</tr>
<tr>
<td>FIPSO</td>
<td>1.35E-02</td>
<td>2.01E-02</td>
<td>1.49E-02</td>
<td>1.53E-02</td>
<td>1.75E-03</td>
<td>1.42E-09</td>
<td>1</td>
<td>‡</td>
</tr>
<tr>
<td>SLPSO</td>
<td>1.06E-02</td>
<td>1.65E-02</td>
<td>1.35E-02</td>
<td>1.36E-02</td>
<td>1.67E-03</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 3. The test error of eight POS-BASED algorithms over 25 independent runs on power load forecasting problem with 50,000 FES.

<table>
<thead>
<tr>
<th>Algorithm/Result</th>
<th>min</th>
<th>max</th>
<th>median</th>
<th>mean</th>
<th>std</th>
<th>p</th>
<th>h</th>
<th>significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSO</td>
<td>1.62E-02</td>
<td>2.69E-02</td>
<td>2.07E-02</td>
<td>2.10E-02</td>
<td>3.11E-03</td>
<td>1.42E-09</td>
<td>1</td>
<td>‡</td>
</tr>
<tr>
<td>BBPSO</td>
<td>1.97E-02</td>
<td>4.82E-02</td>
<td>2.65E-02</td>
<td>3.01E-02</td>
<td>8.80E-03</td>
<td>1.42E-09</td>
<td>1</td>
<td>‡</td>
</tr>
<tr>
<td>CLPSO</td>
<td>3.09E-02</td>
<td>5.61E-02</td>
<td>4.35E-02</td>
<td>4.03E-02</td>
<td>8.42E-03</td>
<td>1.42E-09</td>
<td>1</td>
<td>‡</td>
</tr>
<tr>
<td>QPSO</td>
<td>1.26E-02</td>
<td>2.27E-02</td>
<td>1.72E-02</td>
<td>1.75E-02</td>
<td>2.69E-03</td>
<td>1.42E-09</td>
<td>1</td>
<td>‡</td>
</tr>
<tr>
<td>MRPSO</td>
<td>1.41E-02</td>
<td>3.07E-02</td>
<td>1.81E-02</td>
<td>1.89E-02</td>
<td>3.44E-03</td>
<td>1.42E-09</td>
<td>1</td>
<td>‡</td>
</tr>
<tr>
<td>UPSO</td>
<td>2.18E-02</td>
<td>3.57E-02</td>
<td>2.96E-02</td>
<td>2.88E-02</td>
<td>4.03E-03</td>
<td>1.42E-09</td>
<td>1</td>
<td>‡</td>
</tr>
<tr>
<td>FIPSO</td>
<td>1.32E-02</td>
<td>1.92E-02</td>
<td>1.47E-02</td>
<td>1.48E-02</td>
<td>1.73E-03</td>
<td>1.42E-09</td>
<td>1</td>
<td>‡</td>
</tr>
<tr>
<td>SLPSO</td>
<td>1.06E-02</td>
<td>1.68E-02</td>
<td>1.35E-02</td>
<td>1.37E-02</td>
<td>1.67E-03</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

In order to statistically compare the proposed algorithm with its peers, the statistical tool Wilcoxon’s rank sum test (Wang et al., 2011) at a 0.05 significance level is usually employed to evaluate whether the median fitness values of two sets of obtained results are statistically different from each other. A p-value less than 0.05 means that the performance of two competitive algorithms is statistically different with 95% certainty (h=1), or the performance of two competitive algorithms has no significant difference (h=0). With the purpose of showing the convergence characteristics of the algorithm, the mean training error values in 25 runs at specified checkpoint (0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0)*MAX_FES will be recorded.

Numerical experiments and results

Since SLPSO is proposed for continuous function optimization problems, it is quite natural to apply the SLPSO to FNNs weight training, whose process can be regarded as a hard continuous function optimization problems with missing data or noise pollution. In order to evaluate the performance of SLPSO-trained FNNs, the power load forecasting problem without noise and with noise cases have been considered in this paper. The performance of SLPSO is compared with six other particle swarm algorithms based on FNNs with the structure 7-5-1, including particle swarm optimization (PSO) (Kennedy et al., 1995), comprehensive learning particle swarm optimization (CLPSO) (Liang et al., 2006), quantum-behaved particle swarm optimization (QPSO) (Sun et al., 2004), bare bones particle swarm optimization (BBPSO) (Kennedy et al., 2003), unified particle swarm optimization (UPSO) (Parsopoulos et al., 2004), particle swarm optimization with a moderate-random-search strategy (MRPSO) (Gao et al., 2011), fully informed particle swarm optimization (FIPSO) (Mendes et al., 2004). Each corresponding table presents the experimental results, and the last three rows of each table summarize the comparison results. For clarity, the best results of seven algorithms in comparison in each Table are marked in boldface. ‡, †, and § denote that the performance of SLPSO is better than, worse than, and similar to that of the corresponding algorithm, respectively.

From the statistical results of Table 2 and Table 3, we can note that the SLPSO performs significantly better than other competitors, which is based on the Wilcoxon’s rank sum test results in recording the mean training error and test error.

More specifically, SLPSO outperforms PSO, BBPSO, CLPSO, QPSO, MRPSO, UPSO, FIPSO in both training error and test error results. The experiment results further contend that SLPSO has a good global search ability for FNNs training.

But beyond that, eight evolving curves that compare the performances of eight PSO-based algorithms for the mean training error are shown in Figure 4. These evolving curves display that how SLPSO gradually converges towards the
optimal values faster than the other seven algorithms.

Figure 4. The evolution of the mean training error values derived from eight PSO-based algorithms

Conclusions
A newly proposed meta-heuristic algorithm—SLPSO is first used to find the optimal combination of connection weights and biases for FNNs by converting the problem to that of multi-dimensional continuous optimization problem. Numerical simulation and comparison results on a case study on power load forecasting problem show that SLPSO is more effective with respect to other seven state-of-the-art PSO-based algorithms. Moreover, it further shows that SLPSO can be a powerful tool for optimization other practical optimization problems in other related areas.

Acknowledgments
This work is partly supported by the National Natural Science Foundation of China under Project Code (61561001, 61462002), Beifang University of Nationalities under Project Code (21500080).

References


