Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 6811401, 18 pages
https://doi.org/10.1155/2022/6811401

Research Article

An Improved Self-Organizing Migration Algorithm for Short-Term Load Forecasting with LSTM Structure Optimization

Xiaofeng Rong, Hanghang Zhou, Zijian Cao, Chang Wang, and Linjuan Fan

School of Computer Science and Engineering, Xi'an Technological University, Xi'an 710021, China

Correspondence should be addressed to Zijian Cao; bosscao@163.com

Received 6 October 2022; Revised 13 December 2022; Accepted 14 December 2022; Published 26 December 2022

Academic Editor: Junwei Ma

Establishing an accurate and robust short-term load forecasting (STLF) model for a power system in safe operation and rational dispatching is both required and beneficial. Although deep long short-term memory (LSTM) networks have been widely used in load forecasting applications, it still has some problems to optimize, such as unstable network performance and long optimization time. This study proposes an adaptive step size self-organizing migration algorithm (AS-SOMA) to improve the predictive performance of LSTM. First, an optimization model for LSTM prediction is developed, which divides the LSTM structure seeking into two stages. One is the optimization of the number of hidden layer layers, and the other optimizes the number of neurons, time step, learning rate, epochs, and batch size. Then, a logistic chaotic mapping and an adaptive step size method were proposed to overcome slow convergence problems and stacking into local optimum of SOMA. Comparison experiments with SOMA, PSO, CPSO, LSOMA, and OSMA on test function sets show the advantages of the improved algorithm. Finally, the AS-SOMA-LSTM network prediction model is used to solve the STLF problem to verify the effectiveness of the proposed algorithm. Simulation experiments show that the AS-SOMA exhibits higher accuracy and convergence speed on the standard test function set and has strong prediction ability in STLF application with LSTM.

1. Introduction

Electric load forecasting has a great impact on dispatching work and production scheme of the power system [1, 2]. Accurate STLF is not only necessary for the power grid's steady and safe functioning but also provides significant economic benefits to power corporations [3, 4]. The research shows that the prediction error is reduced by 1%, which can save 1.6 million dollars per year for a 10 GW power plant [5]. Moreover, as the key to intelligent power system research, intelligent load forecasting is of great significance for promoting the construction of smart city in the future [6]. The STLF problem is to predict future power consumption by analyzing the power consumption in the past period [7]. The traditional forecasting method analyses the chart, which is greatly affected by the weather and has a low degree of accuracy. With the development of statistical software and artificial intelligence technology, several prediction methods with higher accuracy have appeared. It mainly includes the time series method, autoregressive integral moving average (ARIMA), and so on [8]. This method's basic idea is to use the temporal nature of the historical load data to forecast. So, it has a higher prediction accuracy for the data with solid timing. But its predictive ability is limited for data with many nonlinear relationships. With the development of power systems, the amount of data is so large that its nonlinear relationships become more complex. Intelligent algorithms are mainly machine learning methods represented by the support vector machine (SVM) [9], random forest (RF) [10], and artificial neural network (ANN) [11]. Most of these algorithms require setting the time feature manually, and the data temporal correlation feature must be fully considered. For long-term data series, their predictive ability is limited.

The deep LSTM network is widely used in STLF problems [12], which is the latest time series forecasting model. According to their composition, methods for using an LSTM network to solve STLF problems are classified as the mixed model (M-model) or optimization

model (O-model). An LSTM network is used in the M-model to learn the nonlinear relationship after extracting features of STLF data. For example, the CNN-LSTM [13] model mixes the convolutional neural network (CNN) operated by three one-dimensional convolution pools and the three-layer LSTM. The model composed of CNN without a pooling layer [14], deep LSTM, and the CNN-SEQ2SEQ-ATT [15] model of attention mechanism. M-model predictions have been proven to be more effective than LSTM alone. The O-model uses an optimization algorithm to adjust the network structure of LSTM, which can achieve more accurate load prediction. In addition, the optimization algorithm is mainly represented by evolutionary computation (EC), which includes IBA-LSTM [16], IGOA-LSTM [17], and EMD-PSO-LSTM [18]. The original evolutionary algorithm is usually optimized to overcome its flaw of falling into local optimal in the previous models. The convergence speed and convergence accuracy are also improved to get a more accurate load prediction effect. Nowadays, deep LSTM neural networks are still facing numinous challenges. For example, network performance is unstable and parameter optimization takes a long time [19, 20].

EC and its variants have proposed many solutions to the abovementioned NP-hard problems, such as genetic algorithm (GA), differential evolution (DE), particle swarm optimization (PSO), artificial bee colony (ABC), and Grey Wolf optimization (GWO) [21]. Parameter optimization relies on employee experience in the LSTM model for flood prediction based on rainfall-runoff observation data [22]. PSO was proposed to optimize the batch size, time step, and number of hidden layer neurons of the LSTM. The comparison experiment proves that the PSO-LSTM model can effectively improve flood prediction accuracy over different advanced periods. In prediction shear wave velocity (Vs) [23], an adaptive subgroup division and particle updating method were proposed to optimize PSO to avoid the PSO-LSTM model falling into local optimization. Then, the batch size, time step, and the number of hidden layer neurons in the LSTM are optimized iteratively by using the optimized PSO. Compared with other algorithms, this method has higher prediction accuracy and robustness. These methods have achieved good results. However, due to their complexity, it is not easy to maintain a balance between convergence speed, convergence accuracy, and robustness [24, 25].

As a classic EC, the self-organizing migrating algorithm (SOMA) [26] was proposed by Zelinka and Lampinen and has good performance in engineering application [27]. SOMA has the advantages of easy parallelization and fewer tuning parameters than others. It has been widely used in dynamic constraint optimization [28], path planning [29], image processing [27], and other fields. At present, the improvement of SOMA has achieved a sound effect, which has provided an excellent theoretical basis for its application in the engineering field. SOMGA [30] is a combination of GA and SOMA. The initial phase of SOMGA uses tournaments to compete against each other, creating new individuals through single-point crossing and bit mutation. This algorithm has stronger robustness than GA and SOMA in 25 test functions. mNM-SOMA [31] uses the NM crossover operator to find the optimal leader, and compared with SOMA, GA, and PSO, the performance of mNM-SOMA is the most superior. The CCMA-ES-SOMA [28] model uses the CCMA-ES algorithm to detect the feasible region of the optimization problem quickly. SOMA has obvious advantages in solving NP-hard problems. However, there needs to be literature on structural optimization of using neural networks. Therefore, a SOMA improvement scheme was proposed to solve the LSTM structure optimization problem in this study.

2. Materials and Methods

2.1. LSTM and Parameter Optimization

2.1.1. LSTM Memory Unit. LSTM solves the gradient explosion and dispersion problems of RNN. The subsequent nodes become weaker in perceiving the previous nodes, and it appears that they forget the previous information as time passes when the number of network layers increases. In short, LSTM can perform better in longer sequences than a normal RNN. Figure 1 shows the operation of the LSTM memory unit. Compared with RNN, LSTM adds a memory unit specifically for saving historical information. The control of input, forget, and output gates updates the history information in the network.

The forward propagation process of LSTM can be expressed as formulas (1)–(5), where $W_{xf}, W_{hf}, W_{xi}, W_{hi}, W_{xo}, W_{ho}$ is the weight matrix, the b_i, b_f, b_c, b_o is the bias corresponding to the weights, the σ is sigmoid function, and $\odot$ is the matrix dot product operation. The input and output vectors of the implicit layer of the LSTM are x_t and h_t at a time step of t. The memory unit is c_t and the input gate is used to control that how much of the network's current input data x_t flows into the memory cell, in other words, how much can be saved to c_t, and the values are expressed as follows:

$$i_t = \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + b_i\right). \tag{1}$$

The forget gate is a key component of the LSTM unit, which controls information to retain and forget. It uses somehow to avoid the gradient disappearance and explosion problems triggered when the gradient propagates backwards in time. The forget gate determines what historical information will be discarded. The information in the memory cell c_{t-1} of the previous moment has an impact on the current memory cell c_t.

$$f_t = \sigma\left(W_{xf}x_t + W_{hf}c_t + b_f\right), \tag{2}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tan h\left(W_{xc}x_t + W_{hc}h_{t-1} + b_c\right). \tag{3}$$

The output gate controls the effect of the memory cell c_t on the current output value h_t, and the part of the memory cell will be printed at the time step of t.

$$o_t = \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + b_o\right), \tag{4}$$

$$h_t = o_t \odot \tan h\left(c_t\right). \tag{5}$$

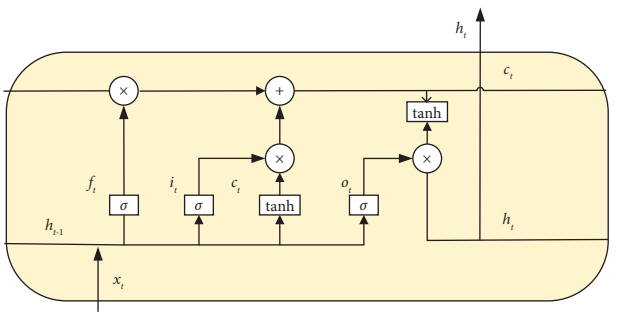


Figure 1: Schematic diagram of LSTM memory cell.

Because of its excellent performance, LSTM is used for a large number of sequence learning tasks, such as robot control [32], speech recognition [33], time series prediction [34, 35], and market prediction [36, 37].

2.1.2. Deep LSTM. The classical LSTM model is composed of an input layer, hidden layer, and output layer. A deep LSTM can be formed by stacking multiple (≥2) hidden layers.

Each layer solves a portion of the task before passing it on the next layer, until the final layer provides the output. Graves et al. constituted a deep LSTM by stacking LSTM hidden layers [38]; it is applied to the speech recognition problem which has obvious advantages in benchmark tests. The deep LSTM architecture is defined as a model that consists of multiple LSTM layers. The upper LSTM hidden layer provides sequential output to the lower layer instead of outputting a single value. This shows that when building the model, the depth of the network is more important than the number of LSTM cells in a given layer. The structure of the time-expanded deep LSTM recurrent neural network is shown in Figure 2.

2.1.3. Optimization of Deep LSTM Parameters. The LSTM recurrent neural network is a stable technique for time series processing, and the number of hidden layers in the LSTM changes the abstract value of the input observation. This method increases the training time and memory cost exponentially. Meanwhile, the disappearance of interlayer gradient will lead to the weakening of network performance, and this phenomenon becomes more significant when there are many layers. That will lead to slow update iterations in the hidden layers which is closer to the input layer and the effectiveness and efficiency of convergence will decline sharply, and will even have easier access to local minima. As shown in Figure 3, the input to the LSTM is a three-dimensional vector consisting of batch size, time step, and features, which are described below.

The current mainstream training method for DNN is gradient descent, which is trained using the model inputs for prediction. Then, the predicted values are compared with the actual values as an estimate of the error, the specified loss function is used to update the model weights, and the process is repeated. Each time step is the feature data that input to the model each time, whose size determines the overall size of the model's single input features. In addition, it also affects the mapping of the model's prediction results to times.

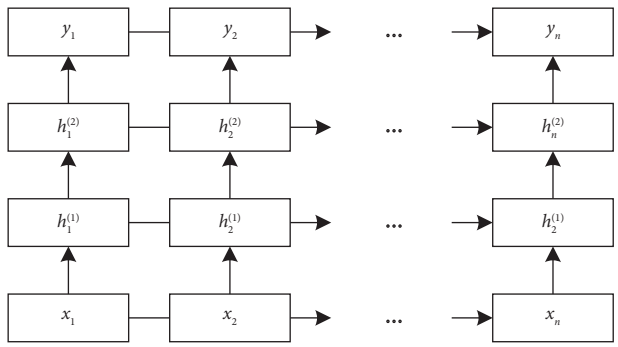


Figure 2: Structure of the deep LSTM recurrent neural network.

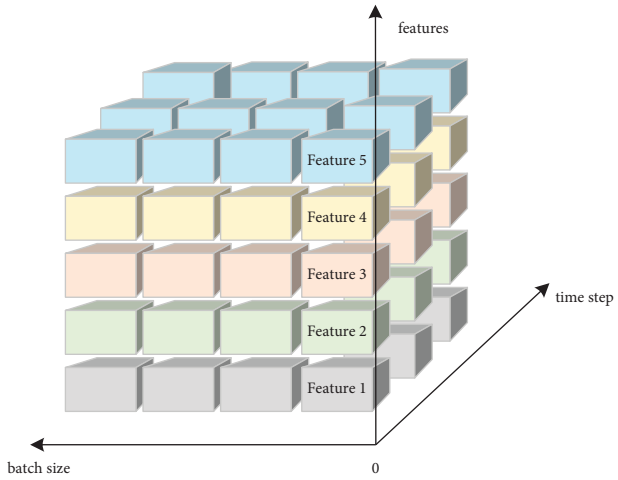


Figure 3: Schematic diagram of the input three-dimensional vector of the LSTM.

The network model weights are updated based on a subset of the training data, which is set in batches. Batch size is the number of samples used to estimate the error gradient in the training data set, in which a sample is the set of eigenvalues at one of the abovementioned time steps. As a statistical estimation, the more training data for the error gradient and the more estimates are computed; the more likely the weights of the network to be better tuned, the better the model's performance. Thus, if the model is used for more predictions, the cost of error gradient estimation is constantly improved. Batch size is an important hyperparameter that affects the learning algorithm's dynamics.

The learning rate is a hyperparameter used by the network model. It controls the weight degree of the updating model according to the estimation error and the speed at which the model adapts to the problem. Lower learning rates require more training cycles for adequate training, while more significant learning rates lead to rapid changes in network model weights and require fewer training cycles. However, if the learning rate is too high, the model may converge to the suboptimal solution quickly, while if the learning rate is too low, the process may be stagnant.

Epochs defines the number of times the learning algorithm is trained over the entire training data set. Iteration, which

consists of one or more batches, means that each sample in the training dataset has completed one prediction of the model. The number of iterations directly affects the number of features learned by the model. Too much or too little of it will result in overfitting and underfitting of the trained model.

2.2. AS-SOMA Model

2.2.1. SOMA. Three main phases of the SOMA are as follows: initialization, migration cycle, and end of migration. In the first stage, each particle of the initial population finds the corresponding fitness value according to the specific problem and determines the best leader. The key to the SOMA is that the second phase of the migration cycle composes multiple cumulative migration updates. The particle repeatedly makes small jumps to the leader by adopting a specific step size and randomly initializing the guidance of the perturbation PRTVrctor. It is a constraint variable that controls the particle's movement dimension, and the expression is defined as follows:

$$\text{PRTVrctor} = \begin{cases} 1, & \text{if rand} < \text{prt}, \\ 0, & \text{else}, \end{cases} \tag{6}$$

where rand is a random value between [0, 1], and PRTVrctor is regulated by setting the coefficient prt = [0, 1]. The mode of particle migration can be expressed as follows

$$X_{\text{temp}}^{ML} = X_{i,\text{state}}^{ML} + \left(X_L^{ML} - X_{i,\text{state}}^{ML}\right) * t_i^{ML} * \text{PRTVrctor}, \tag{7}$$

where $X_{i,\text{state}}^{ML}$ and X_L^{ML}, respectively, represent the particle to be migrated and the leader in the migration cycle of the first ML generation and X_{temp}^{ML} is the new position obtained by the particle migration. t_i^{ML} is expressed as the interval length of step as follows:

$$t_i^{ML} = t_{i-1}^{ML} + \text{step}. \tag{8}$$

The cycle will be stopped when the migration of particles reaches the accumulated maximum PathLength. Meanwhile, the selected leader will lead the particles to carry out the next round of migration and the end of migration when the algorithm stop condition is met.

2.2.2. Shortcomings of SOMA. SOMA has significant flaws in the initialization and migration cycle phases due to its design principles. For example, the initialization scheme and update mode used the idealized randomness and the fixed step, which has problems with slow convergence and easy to fall into local optimal in optimization. The following is a detailed introduction.

To follow the law of biological population, SOMA adopts a random scheme in the initialization stage. However, this random initialization scheme may lead to problems, such as the particle is only in the local optimal region, the population density is too large, and the individual spacing needing to be more prominent in the optimization process. Then, the slow convergence, local optimum, and advance convergence will occur in the optimization process.

In the migration cycle stage, due to the fixed value of the migration step, the particles cannot migrate fully and effectively. Large or small steps affect the execution efficiency and convergence ability of the algorithm to varying degrees. The selection method of SOMA for the leader of each migration is the maximum fitness, which can effectively guide the population towards a better direction of migration. But it needs to include the diversity of guidance and may even lead to incorrect guidance at the initialization stage of algorithm execution when the leader is locally optimal.

2.2.3. AS-SOMA. The logistic chaotic mapping and the adaptive step size method are used to optimize SOMA in the initialization stage and migration cycle stage, respectively, in this paper. The particles can be evenly distributed in the whole search space in the initialization stage, which promotes the balance between development and exploration in the renewal process. Logistic map is one of the simplest chaotic maps, and it is described as formula (9), where c_i represents the ith chaotic number, the $c_i \in (0, 1)$, and μ is an adjustable parameter.

$$c_{i+1} = f(\mu, c_i) = \mu c_i (1 - c_i), i = 0, 1, 2 \ldots. \tag{9}$$

In the initialization stage of AS-SOMA, the multiple chaotic sequence numbers were generated by changing the initialization value of logistic chaotic map c_0. The obtained chaotic number is multiplied by the search space of different dimensions, taking it as a coefficient to obtain the particles evenly distributed in the whole search space. The specific calculation is described as follows:

$$x_{ij} = c_{ij} \times \left(p_{j\max} - p_{j\min}\right), \tag{10}$$

where $p_{j\max}$ and $p_{j\min}$ are upper and lower bounds of particles in the j-dimension, respectively. Assuming that $p_{j\max} = 120$ and $p_{j\min} = 0$, in Figure 4, there is no blind area when $c_0 = 0.6$, $i = 200$, and $\mu = 4$. Therefore, the value of μ is 4 in this document, c_i is evenly distributed in the interval of $(0, 1)$, and the particles in the search space are uniformly distributed in $(0, 120)$.

The adaptive step method is used to optimize the fixed value of the original step in the AS-SOMA migration cycle. The method stores the step size information of successfully migrated particles by creating an archive of success information S_{step}, which is going to accumulate to form the mean of the normal distribution. The next migration of particles is guided by generating random step in the way of normal distribution, which is described as follows:

$$\text{step} = \text{rand}_i(\mu\,\text{step}, 0.05). \tag{11}$$

The μ step is initialized to 0.21 for better results [39] and updated at the end of each migration cycle in this paper, and the updated formula is as follows:

$$\mu\,\text{step} = (1 - c) * \mu\,\text{step} + c * \text{mean}_A\left(S_{\text{step}}\right), \tag{12}$$

where c is a constant between 0 and 1, and S_{step} represents the step size archive of successfully migrated particles.

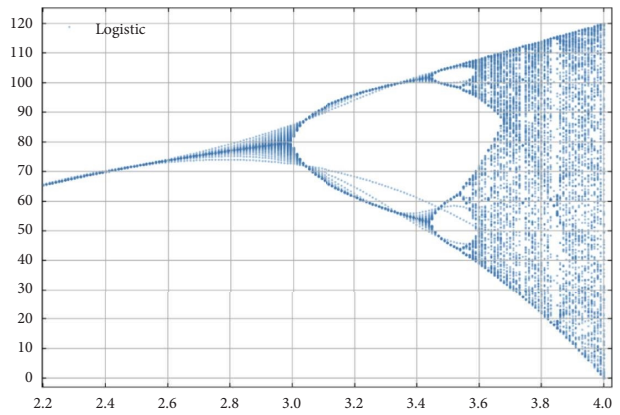


Figure 4: Change of logic mapping with μ.

2.3. AS-SOMA-LSTM Optimization Model

2.3.1. LSTM Training. The training model of depth LSTM is shown in Figure 5. The parameters to be optimized include hidden layer, number of neurons, time step, learning rate, epochs, and batch size. To make the network weights converge more stably during the model's training process, this paper uses exponential descent for the hyperparameter learning rate, which decreases exponentially as the number of LSTM training generations increases. So, the optimization process only searches for its initial value.

2.3.2. STLF Problem Based on the AS-SOMA-LSTM Optimization Model. As shown in Figure 6, the AS-SOMA based on the above LSTM training model encodes the particles as $x_i(h_{i1}, h_{i2}, ts_i, lr_i, mi_i, bs_i)$ where i is the ith particle, h_{i1} and h_{i2} are the number of neurons in the first and second hidden layers, respectively. ts_i is the time step of the model input, lr_i, mi_i, bs_i are the size of the initial learning rate, the epochs of training iterations, and the batch size in model training, respectively. The abovementioned particles were decoded as parameters of the LSTM network model for training and prediction, and the optimal parameters were obtained through continuous iteration according to AS-SOMA.

The STLF problem processing process based on the AS-SOMA-LSTM optimization model is shown in Figure 7. First, the AS-SOMA initializes the population to generate N individuals, which generates the corresponding set vector PRTVrctor by setting the coefficient prt. Subsequently, the dimension parameters are configured in LSTM to construct the corresponding prediction model, which is derived from decoding individuals in the population. The model is trained and tested to calculate the root mean square error (RMSE), which is returned to AS-SOMA as a fitness function value. AS-SOMA sorted the fitness function values of the obtained population and selects the leader with the minimum value to guide the next population migration. The calculation process of population migration is repeated, and the number of repetitions is the maximum number of iterations set by AS-SOMA. The value of the last generation leader is the hyperparameter of the optimal model.

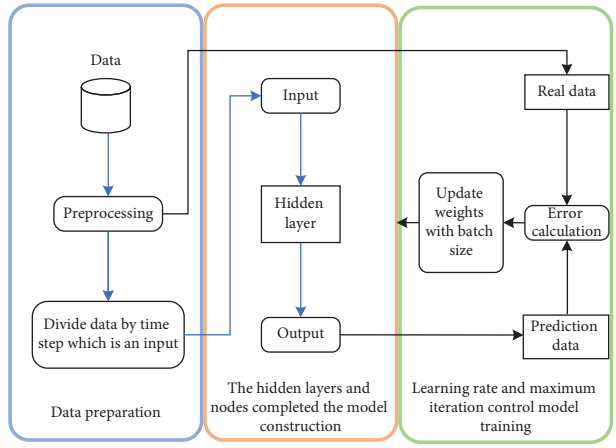


Figure 5: LSTM training model diagram.

3. The Performance of the AS-SOMA

3.1. Experimental Parameters. MATLAB is used to conduct comparative experiments on fifteen test functions of CEC2015 [40] in this paper. According to the basic characteristics, the CEC2015 can be divided into the following four categories: F1 and F2 are single-peak functions, F3, F4, and F5 are basic multipeak functions, F6, F7, and F8 are three mixed functions, and F9–F15 are seven synthetic functions. The experimental comparison of AS-SOMA with the mainstream population intelligence algorithm PSO and its modified algorithm CPSO and the parameter settings of each algorithm are given in Table 1.

3.2. Algorithm Evaluation. In order to improve the convergence accuracy of AS-SOMA, the experiments were evaluated by the mean and standard deviation of the best run results on the functions, and it is described as $f(x) - f(x^*)$. Then, this paper is compared with other algorithms in multiple run results, and the mathematical expression is described as follows:

$$\begin{aligned} \text{Mean} &= \frac{1}{N}\sum_{i=1}^{N} f(x_i) - f(x_i^*), \\ \text{Std}_N &= \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(f(x_i) - f(x_i^*) - \text{Mean}\right)^2}, \end{aligned} \tag{13}$$

where $f(x^*)$ is the global optimal value of the algorithm on the function and i is the number of times the function runs. Wilcoxon's rank-sum test is performed on the obtained experimental data to verify the significance of the performance between individual algorithms. If P value is less than 0.05 ($H = 1$), it indicates that the operating results of the current algorithm are significantly different from those of AS-SOMA. The convergence rate is compared by observing the convergence rate of the fitness value of the benchmark function.

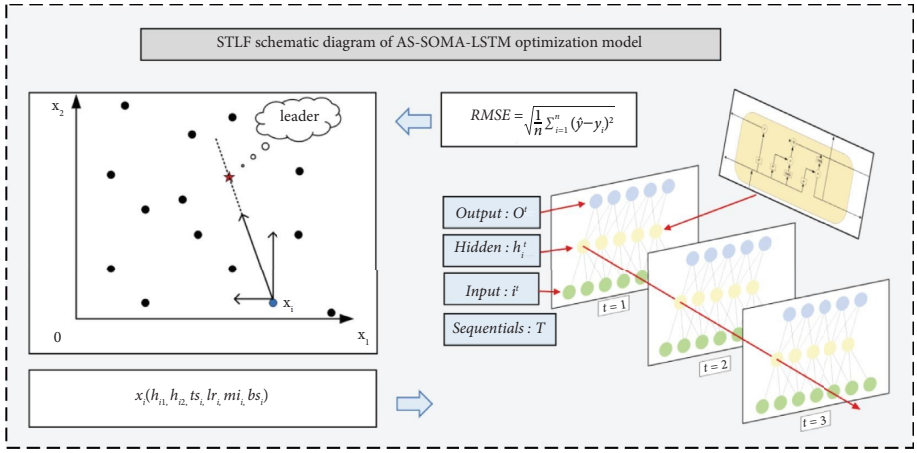


Figure 6: STLF schematic diagram of the AS-SOMA-LSTM optimization model.

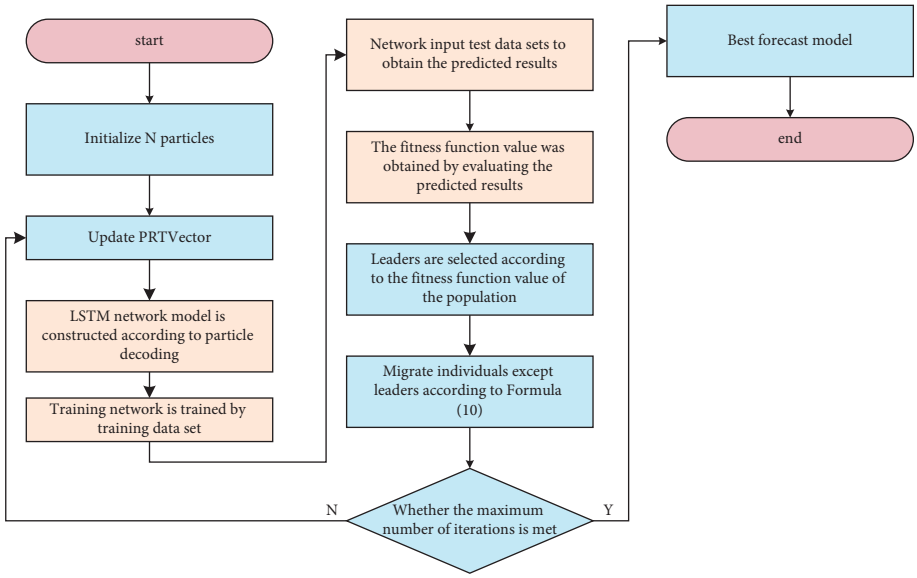


Figure 7: STLF problem flowchart of the AS-SOMA-LSTM optimization model.

Table 1: Parameter settings of the comparison algorithm.

Algorithm	Parameter setting
PSO, CPSO [41]	$w = 0.8, c_1 = c_2 = 2.0$
SOMA	prt = 0.1, PathLength = 2, step = 0.21
LSOMA [42]	$\text{Step}_{\max} = 0.4, \text{Step}_{\max} = 0.7$
OSOMA [43]	prob = 0.5

3.3. Experimental Analysis of AS-SOMA Performance. In the experiment, the population size of PSO, CPSO, SOMA, LSOMA, OSOMA, and AS-SOMA are 100, and the maximum evaluation times of CEC2015 benchmark function is 300,000, among which the times of individual tests of each function is 25 and its dimension $D = 30$.

3.3.1. The Solving Accuracy. Table 2 shows the experimental results of each algorithm on CEC2015 benchmark function set. "+", "=" and "−" indicate that the accuracy of the corresponding algorithm is better, similar, and worse than AS-SOMA. The bold display indicates the best result of the mean and the best variance.

AS-SOMA achieves optimal experimental results on both single-peak functions, which are compared with the five comparison algorithms as shown in Table 2. Wilcoxon's rank-sum test results show that the performance of AS-SOMA is significantly better than other comparison algorithms on function F_2. In the test results of function F_1, AS-SOMA has a similar convergence performance with SOMA, and it is significantly better than 4 of the five comparison algorithms. Compared with basic multimodal functions, AS-SOMA achieved two best results in three functions. Wilcoxon's rank-sum test results showed that AS-SOMA is significantly better than PSO, CPSO, SOMA, LSOMA, and OSOMA on the test functions of (F_4, F_5), (F_4, F_5), (F_3), (F_3, F_4 and F_5), and (F_3, F_5). In the mixed function, it is obvious that the experimental results of AS-SOMA on function F_6 are better than other algorithms. It is obvious that the experimental results of AS-

Table 2: Precision comparison results of fifteen benchmark function solutions in the CEC2015.

Functions	Results	PSO	CPSO	SOMA	LSOMA	OSOMA	AS-SOMA
F_1	Mean	2.10 $E+07-$	1.83 $E+08-$	4.71 $E+05=$	9.75 $E+05-$	6.03 $E+05-$	**3.34 $E+05$**
	Std	3.97 $E+07$	1.91 $E+08$	4.11 $E+05$	5.45 $E+05$	4.20 $E+05$	3.99 $E+05$
	P value	1.58 $E-02$	7.16 $E-05$	2.64 $E-01$	9.67 $E-06$	2.42 $E-02$	—
	H value	1	1	0	1	1	—
F_2	Mean	3.54 $E+09-$	7.39 $E+09-$	3.60 $E+03-$	2.80 $E+03-$	3.12 $E+03-$	1.95 $E+03$
	Std	2.58 $E+09$	4.96 $E+09$	3.18 $E+03$	3.16 $E+03$	3.65 $E+03$	2.50 $E+03$
	P value	4.24 $E-07$	1.08 $E-07$	7.51 $E-03$	3.40 $E-02$	1.64 $E-02$	—
	H value	1	1	1	1	1	—
F_3	Mean	2.06 $E+01=$	**2.04 $E+01+$**	2.08 $E+01-$	2.08 $E+01-$	2.08 $E+01-$	2.07 $E+01$
	Std	2.47 $E-01$	2.04 $E-01$	3.52 $E-02$	5.69 $E-02$	4.14 $E-02$	1.17 $E-01$
	P value	1.09 $E-01$	3.02 $E-08$	4.15 $E-03$	1.79 $E-06$	6.73 $E-03$	—
	H value	0	1	1	1	1	—
F_4	Mean	7.40 $E+01-$	8.34 $E+01-$	3.73 $E+01=$	4.55 $E+01-$	3.42 $E+01=$	**3.78 $E+01$**
	Std	2.53 $E+01$	2.02 $E+01$	8.49 $E+00$	1.28 $E+01$	9.29 $E+00$	1.09 $E+01$
	P value	3.60 $E-08$	8.88 $E-10$	8.62 $E-01$	1.33 $E-02$	2.56 $E-01$	—
	H value	1	1	0	1	0	—
F_5	Mean	2.74 $E+03-$	4.30 $E+03-$	1.93 $E+03=$	5.13 $E+03-$	2.17 $E+03-$	**1.67 $E+03$**
	Std	4.95 $E+02$	5.93 $E+02$	7.73 $E+02$	9.14 $E+02$	1.05 $E+03$	6.08 $E+02$
	P value	1.89 $E-07$	3.65 $E-13$	1.86 $E-01$	6.54 $E-15$	1.88 $E-02$	—
	H value	1	1	0	1	1	—
F_6	Mean	5.81 $E+05-$	6.37 $E+06-$	2.25 $E+05-$	3.66 $E+05-$	1.89 $E+05=$	**1.43 $E+05$**
	Std	1.01 $E+06$	1.14 $E+07$	1.69 $E+05$	3.02 $E+05$	1.83 $E+05$	9.34 $E+04$
	P value	3.66 $E-02$	1.13 $E-02$	2.83 $E-02$	1.72 $E-03$	2.46 $E-01$	—
	H value	1	1	1	1	0	—
F_7	Mean	2.26 $E+01-$	5.03 $E+01-$	**8.87 $E+00=$**	1.08 $E+01=$	9.43 $E+00=$	9.81 $E+00$
	Std	9.14 $E+00$	6.08 $E+01$	1.96 $E+00$	2.21 $E+00$	1.88 $E+00$	1.41 $E+00$
	P value	3.12 $E-07$	2.90 $E-03$	5.52 $E-02$	7.22 $E-02$	3.39 $E-01$	—
	H value	1	1	0	0	0	—
F_8	Mean	4.85 $E+04=$	8.56 $E+05-$	4.53 $E+04-$	9.18 $E+04-$	5.08 $E+04-$	**2.70 $E+04$**
	Std	8.03 $E+04$	7.77 $E+05$	3.05 $E+04$	4.04 $E+04$	3.60 $E+04$	2.02 $E+04$
	P value	2.13 $E-01$	1.83 $E-05$	1.98 $E-02$	4.86 $E-08$	2.40 $E-03$	—
	H value	0	1	1	1	1	—
F_9	Mean	1.46 $E+02-$	1.52 $E+02-$	**1.03 $E+02-$**	**1.0 $E+02-$**	**1.03 $E+02-$**	**1.03 $E+02$**
	Std	4.28 $E+01$	3.57 $E+01$	**1.80 $E-01$**	2.18 $E-01$	1.91 $E-01$	3.14 $E-01$
	P value	2.90 $E-05$	3.82 $E-07$	1.13 $E-02$	2.59 $E-06$	2.28 $E-03$	—
	H value	1	1	1	1	1	—
F_{10}	Mean	5.45 $E+05=$	2.31 $E+06-$	7.57 $E+04=$	1.41 $E+05-$	8.93 $E+04=$	**6.69 $E+04$**
	Std	1.30 $E+06$	3.26 $E+06$	5.33 $E+04$	8.29 $E+04$	7.67 $E+04$	6.47 $E+04$
	P value	8.08 $E-02$	2.26 $E-03$	5.82 $E-01$	4.86 $E-03$	2.95 $E-01$	—
	H value	0	1	0	1	0	—
F_{11}	Mean	8.05 $E+02-$	8.75 $E+02-$	4.79 $E+02-$	5.28 $E+02-$	4.93 $E+02-$	**3.20 $E+02$**
	Std	2.66 $E+02$	4.52 $E+02$	1.75 $E+02$	1.73 $E+02$	1.77 $E+02$	7.56 $E+01$
	P value	4.43 $E-09$	1.62 $E-06$	7.00 $E-04$	3.85 $E-05$	5.34 $E-05$	—
	H value	1	1	1	1	1	—
F_{12}	Mean	1.28 $E+02-$	1.43 $E+02-$	**1.06 $E+02-$**	1.07 $E+02-$	**1.06 $E+02-$**	**1.06 $E+02$**
	Std	1.47 $E+01$	1.25 $E+01$	**4.51 $E-01$**	6.73 $E-01$	9.47 $E-01$	6.42 $E-01$
	P value	1.19 $E-07$	1.27 $E-13$	1.08 $E-02$	1.01 $E-07$	2.70 $E-02$	—
	H value	1	1	1	1	1	—
F_{13}	Mean	1.22 $E+02-$	1.43 $E+02-$	1.06 $E+02-$	1.17 $E+02-$	1.05 $E+02-$	**1.01 $E+02$**
	Std	7.55 $E+00$	7.18 $E+00$	4.89 $E+00$	3.06 $E+00$	4.16 $E+00$	7.14 $E+00$
	P value	4.03 $E-11$	1.95 $E-16$	1.26 $E-02$	5.93 $E-11$	1.50 $E-02$	—
	H value	1	1	1	1	1	—
F_{14}	Mean	4.26 $E+04-$	4.39 $E+04-$	3.30 $E+04-$	3.34 $E+04-$	3.33 $E+04-$	**3.20 $E+04$**
	Std	5.03 $E+03$	6.98 $E+03$	1.12 $E+3$	9.31 $E+02$	1.16 $E+03$	7.73 $E+02$
	P value	1.29 $E-10$	5.55 $E-09$	5.55 $E-04$	7.59 $E-08$	2.76 $E-04$	—
	H value	1	1	1	1	1	—

Table 2: Continued.

Functions	Results	PSO	CPSO	SOMA	LSOMA	OSOMA	AS-SOMA
F_{15}	Mean	$1.25\,E+02-$	$2.61\,E+02-$	$\mathbf{1.00\,E+02=}$	$1.00\,E+02-$	$\mathbf{1.00\,E+02=}$	$\mathbf{1.00\,E+02}$
	Std	$7.37\,E+00$	$3.33\,E+02$	$1.58\,E-13$	$1.51\,E-10$	$\mathbf{1.45\,E-13}$	$3.99\,E-13$
	P value	$8.52\,E-15$	$2.36\,E-02$	$3.56\,E-01$	$1.31\,E-09$	$3.35\,E-01$	—
	H value	1	1	0	1	0	—
−/+/=		12/0/3	14/1/0	9/0/6	14/0/1	10/0/5	—

The bold values display the best result of the mean and the best variance.

SOMA on function F_6 are better than CPSO, SOMA, LSOMA, and OSOMA. It is significantly better than PSO, CPSO, SOMA, and LSOMA on the function F_8. In function F_7, AS-SOMA has some competitiveness compared with basic SOMA in obtaining optimal results. For the test results of F_9, F_{11}, F_{12}, F_{13}, and F_{14} functions of the composite, AS-SOMA is obviously superior to all comparison algorithms.

To sum up, AS-SOMA can solve the synthesis function effectively. The excellent results are mainly attributed to the step size adaptive mechanism and the population initialization mechanism based on the logistic chaotic mapping. The interaction between the two mechanisms improves the solving accuracy of the algorithm.

3.3.2. The Convergence Speed. To further verify the optimization effect of the algorithm in the search process, Figure 8 shows the average convergence result of the comparison algorithms in 15 benchmark functions based on the CEC2015 data set 25 times. The ordinate is the natural logarithm of the average value of the independent 25-times running results of each algorithm. The horizontal coordinate represents the sampling point, and its value is from $FES = 1000$ and $\mathrm{mod}\,(FES, 10000) = 0$.

It can be seen from the experimental results in Figure 8, AS-SOMA achieves peak performance on F_1, F_2, F_5, F_6, F_8, F_{10}, F_{11}, F_{13}, and F_{14} while being competitive on F_4, F_7, F_9, F_{12}, and F_{15}. In addition, CPSO has the best convergence speed on F_3, OSOMA has the best convergence speed on F_4, and SOMA has the best convergence speed on F_7. However, the convergence speed of PSO and LSOMA is not optimal on all benchmark functions.

The experimental results in Figures 8(e) and 8(k) show that the AS-SOMA has excellent convergence speed compared with other comparison algorithms. In addition to the experimental data in Table 2, it can be seen that AS-SOMA has excellent performance on F_5 and F_{11} both in terms of solving accuracy and convergence speed. It can be concluded that AS-SOMA has good convergence in solving basic multimode functions and synthetic functions.

3.3.3. Search Behavior Analysis. In order to explain the convergence performance of AS-SOMA, this paper analyzes the search behavior of AS-SOMA by analyzing the population evolution of AS-SOMA on function F_4. Schwefel′s function has a global minimum point that is far from another local optimum, and it is a typical cheating problem. So, it is hard to get out of local optimum. Figure 9 shows the search process performed by AS-SOMA on F_4 when the population size is 100 and the evolutionary algebra G is 1, 3, 5, and 10, respectively. It is obvious that Schwefel′s function has a very complex model structure. The results of the three-generation evolutionary search performed by the algorithm are shown in Figure 9(b) when the population with a population size of 100 is initialized uniformly and randomly in the solution space. It can be observed that the population migrates toward the global optimum while maintaining diversity.

Figure 10 shows the migration step change diagram of AS-SOMA individuals in the search process of Schwefel′s function, and abscissa represents the evolution time. It is easy to see that step decreases as the number of evaluations increases. The population explores the search space in the early stages of algorithmic search tentatively, the less successful step information in the archive makes less population learnable information at this time, the larger step ensures the search speed of the algorithm. After a period of searching, the success information is increased gradually in the archive. Step begins to decrease steadily through the use of adaptive regulatory mechanisms. It reduces the probability of particles falling into the local optimality.

4. Structural Optimization of LSTM by AS-SOMA

4.1. STLF Data. This paper selects data in minutes from the UCI machine learning warehouse. It collected measurements of electricity consumption in various parts of a household from December 2006 to November 2010. The observed values include total active power (TAP) (kW), total reactive power (TRP) (kW), average voltage (AV) (V), average current (AC) (A), kitchen power (KP) (kWh), washing machine power (WMP) (kWh), and air conditioning system power (ASP) (kWh). This paper uses the replacement of null outliers and data accumulation to preprocess the total active power original data. As shown in Table 3, the statistical unit is expanded from hours to days.

The target problem is the prediction of daily electricity load in the next week. In addition, this paper focuses on the prediction of the TAP, which is the sum of the power consumption of the appliance [44–46]. The 3 : 1 scale partition dataset is presented in this study in Figure 11(a), and the first three years and the fourth year of data completed the

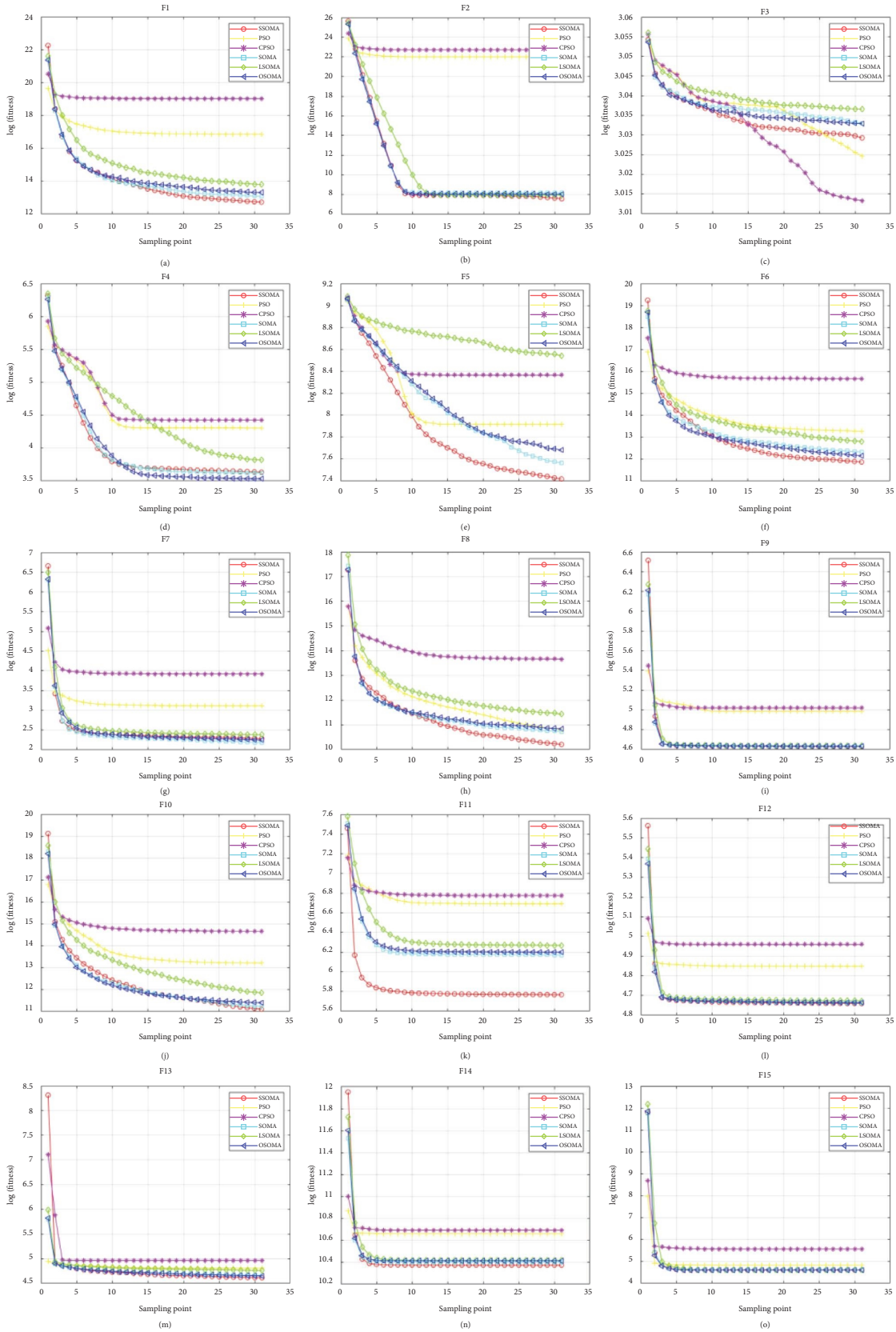


FIGURE 8: Comparison results of convergence rates of solutions of 15 reference functions (F_1–F_{15}) of CEC2015 ($D = 30$). (a) F_1. (b) F_2. (c) F_3. (d) F_4. (e) F_5. (f) F_6. (g) F_7. (h) F_8. (i) F_9. (j) F_{10}. (k) F_{11}. (l) F_{12}. (m) F_{13}. (n) F_{14}. (o) F_{15}.

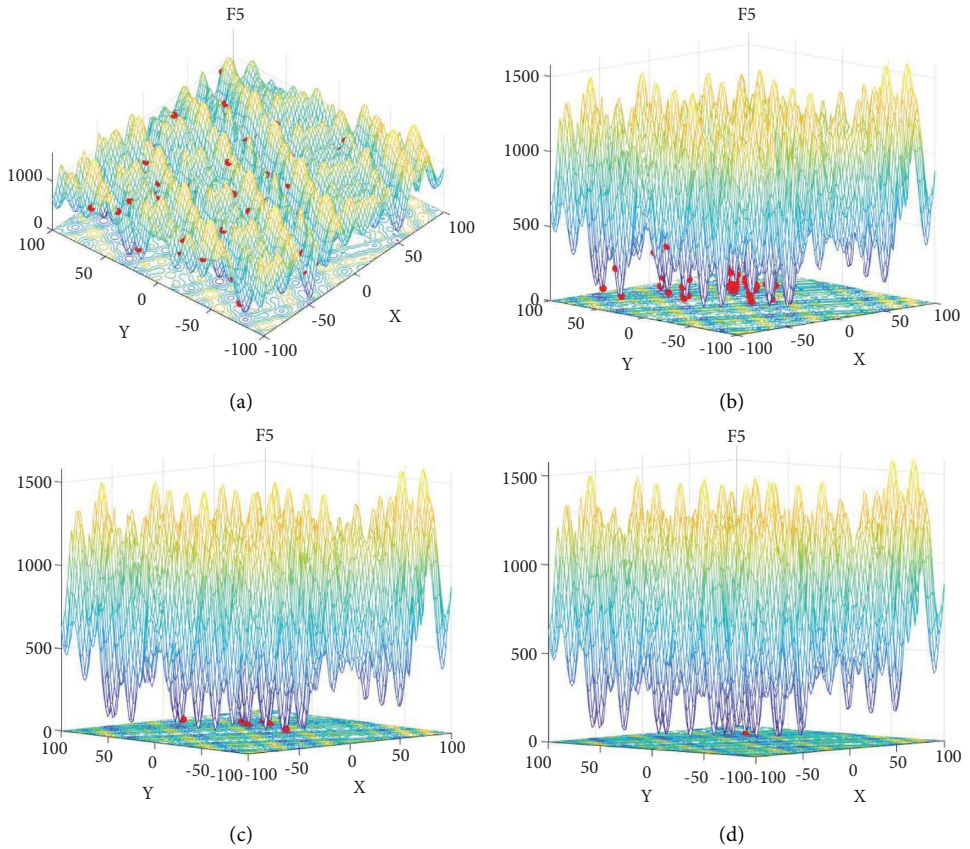


Figure 9: AS-SOMA search process on Schwefel′s function. (a) ($G = 1$). (b) ($G = 3$). (c) ($G = 5$). (d) ($G = 10$).

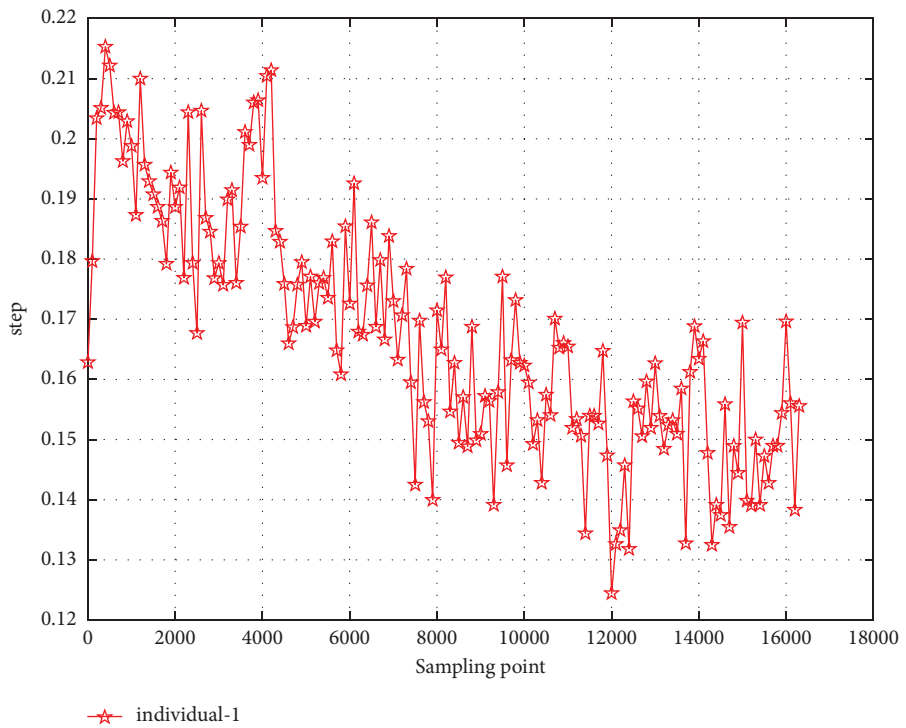
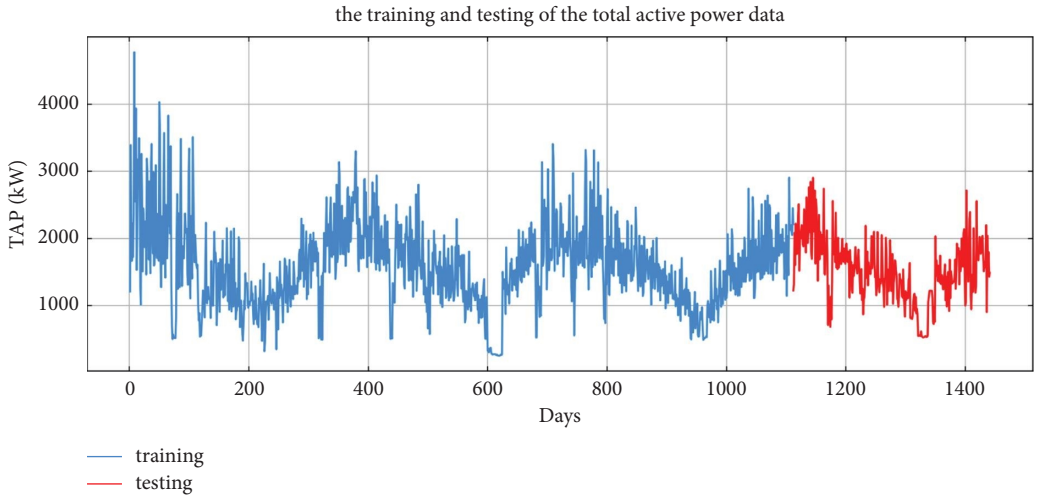


Figure 10: Variation diagram of STEP of AS-SOMA search process on Schwefel′s function.

Table 3: STLF data in days.

Date	TAP	TRP	AV	AC	KP	WMP	ASP	Others
2006/12/16	1209.176	34.922	93552.53	5180.8	0	546	4926	14680.93
2006/12/17	3390.46	226.006	345725.3	14398.6	2033	4187	13341	36946.67
2006/12/18	2203.826	161.792	347373.6	9247.2	1063	2621	14018	19028.43
2006/12/19	1666.194	150.942	348479	7094	839	7602	6197	13131.9
2006/12/20	2225.748	160.998	348923.6	9313	0	2648	14063	20384.8
2006/12/21	1723.288	144.434	347096.4	7266.4	1765	2692	10456	13808.47



(a)

Season	descriptive statistical analysis of the TAP						
	Max	Min	Mean	Range	Std	Skewness	Kurtosis
all	4773.386	250.298	1567.840	4523.088	597.307	0.578	1.278
training	4773.386	250.298	1580.847	4523.088	627.613	0.605	1.194
testing	2903.014	524.746	1524.006	2378.268	479.587	0.118	0.066

(b)

Figure 11: The total active power data.

training and testing of the model. Figure 11(b) shows the statistical indicators of maximum, minimum, mean, median, range, standard deviation, skewness, and kurtosis to conduct a descriptive statistical analysis of the TAP.

4.2. Model Evaluation. The mean squared error (MSE), root mean squared error (RMSE), mean absolute error (MAE), and mean absolute percent error (MAPE) are metrics used to evaluate the predictive performance of the models. The MSE, RMSE, MAE, and MAPE are defined as follows:

$$\begin{aligned} \mathrm{MSE} &= \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i), \\ \mathrm{RMSE} &= \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y} - y_i)^2}, \\ \mathrm{MAE} &= \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|, \\ \mathrm{MAPE} &= \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100\%, \end{aligned} \tag{14}$$

where $\hat{y}$ is the predicted value, y_i is the true value, and the value range of RMSE is $(0, +\infty)$.

4.3. Prediction Experiment Results and Analysis. In order to reduce the operation error, each algorithm is independently run ten times to record the maximum, median, minimum, mean, and standard deviation of the execution results of each algorithm. Wilcoxon's rank-sum test is used to compare the performance of the algorithm.

4.3.1. Hidden Layer Optimization. Structure optimization of LSTM is divided into two stages. The hidden layer number has a small search space and only integer, which is the first stage of structure optimization. The optimization of the number of nodes in each hidden layer is the premise of structural optimization. Other hyperparameters set include hidden layer neurons, learning rate, training algebra, batch size, time steps, and the values are 200, 0.001, 500, 16, and 14. The data within fourteen days were used to predict the data for the next 7 days, and Adam optimizer [47] is used to complete the weight update. This paper compared the predicted results to model the optimal number of hidden layer nodes.

On the data prediction of STLF, Table 4 shows the prediction performance of LSTM with different layers. It can be seen that the network has better performance with the hidden layer digit of 2, and there is a significant difference compared with layer 1 and layer 4. As a result, a model with fewer hidden layers will result in insufficient expression ability of the model to the problem. On the contrary, the disappearance of gradient between layers will lead to weakened network performance when there are too many hidden layers. In the process of prediction model optimization mentioned below, the number of hidden layers of LSTM is determined to be two.

4.3.2. Optimization of Other Parameters. The second stage is to optimize other parameters, including the number of neurons in the hidden layer, time step, learning rate, epochs, and batch size. In order to make the model converge more stable in the training process, this paper uses the method of exponential decline for the hyperparameter learning rate.

Table 5 shows the size of search space corresponding to the parameters when AS-SOMA is used to optimize the network model. The mutation probability of GA parameters in the comparison algorithm is 0.001. Parameter settings of others are consistent with Table 1. As a control experimental algorithm, AS-SOMA_1 implements the improvements in the initialization phase but not in the migration phase. The population size of all algorithms is 10, and the algorithm execution termination condition is 500 times of fitness function evaluation.

For the final optimization model of AS-SOMA-LSTM, the network structure with the best effect is that the first layer contains 47 hidden nodes, the second layer contains 86 hidden nodes, the time step is 11, the batch size is 24, the learning rate is 0.03414, and the epochs is 338. The comparison between AS-SOMA-LSTM model prediction results and original data is shown in Figure 12.

Table 6 shows that AS-SOMA achieves the best results on each metric and outperforms the state-of-the-art methods by 0.44%, 0.37%, 0.117%, and 0.005% in terms of mean MSE, mean RMSE, mean MAE, and mean MAPE.

Table 7 shows the statistics of the experimental results of different algorithms in STLF based on the LSTM optimization model on the RMSE. It can be seen that AS-SOMA has achieved the best results in maximum, median, and average among the eight algorithms. AS-SOMA has more advantages than AS-SOMA_1 in terms of average value and standard difference. Wilcoxon's rank-sum test shows that there is no significant difference in the performance between the two algorithms compared with the other five algorithms. This phenomenon shows that the two SOMA improvement schemes contribute equally to the final performance, and the two schemes jointly improve the performance of AS-SOMA.

AS-SOMA ranks fourth out of seven algorithms in the minimum index. One of the reasons is the algorithm's randomness, where each algorithm has a certain probability of obtaining the optimal value of the problem. Secondly, AS-SOMA proposed an adaptive step mechanism based on SOMA, which improved the convergence ability and stability of the algorithm. Meanwhile, it also reduces the exploration ability of the algorithm. OSOMA ranked first in minimum index, which is based on reverse learning mechanism to improve SOMA. The idea of reverse learning is to promote the particle to migrate in the opposite direction of the best particle with a certain probability. This mechanism enhances the exploration ability of the algorithm greatly. But statistics shows that the results of multiple experiments are inconsistent. According to the combination of mean and standard deviation in Table 5, AS-SOMA has obvious advantages in solving STLF problem by LSTM.

Figure 13 is the average convergence curve of eight algorithms on STLF problem with LSTM.

Figure 13(a) shows the poor performance of GA at the initial stage of search. The reason is that the initial fitness function value of GA in a search reaches 1893.852, which is the particle to get a set of poor solutions in the case of random initialization. The model is not trained as it should be when the particle is decoded into the LSTM. It makes the prediction performance of the network poor.

Figure 13(b) is obtained by trimming the original convergence curve. It can be seen that AS-SOMA has a certain advantage in the whole search process. OSOMA based on reverse learning has the worst performance among the eight algorithms, it shows that the scheme based on reverse learning is not suitable for LSTM structure optimization.

Figure 13(c) shows the eight algorithms in the early stage of search. It can be seen that CPSO, A-MsPSO, and AS-SOMA based on logistic chaotic mapping initialization have faster convergence in the early stage of search. In the initialization stage, the whole population is evenly distributed in the entire search space, which ensures the exploration ability and diversity of the population in the early stage.

Table 4: Experimental results of the number of hidden layers comparison RMSE.

Number of hidden layers	1	2	3	4
Minimum	**360.905**	364.865	365.058	366.565
Median	382.118	**376.758**	379.291	379.532
Median	443.417	**421.458**	483.625	488.773
Average	387.8803	**379.0596**	383.9846	391.2938
Standard deviation	21.89808	**11.49589**	22.84171	31.98162
P value	0.04148917	—	0.171076222	0.040966763
H value	1	—	0	1

The bold value is the smallest RMSE metrics value of each item, which represents the experimental results with good performance.

Table 5: Search space.

Hyperparameter	Search space	Type
Number of nodes at the hidden layer	[1, 200]	Integer
Time step	[1, 14]	Integer
Learning rate	[0.001, 0.01]	Float
Epoch	[1, 500]	Integer
Batch size	[1, 128]	Integer

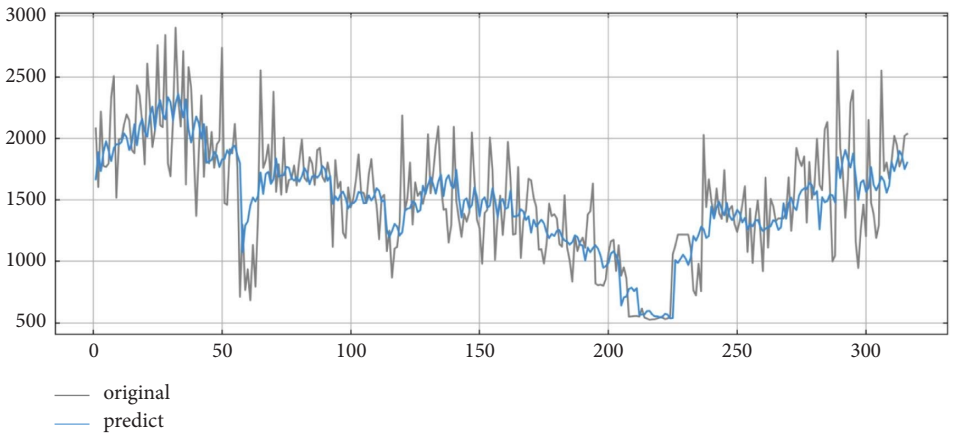


Figure 12: Comparison of AS-SOMA-LSTM prediction results with raw data.

Table 6: Average evaluation results of different models in all evaluation items.

Model	Mean MSE	Mean RMSE	Mean MAE	Mean MAPE
GA	158887.368	358.828	298.243	22.541
PSO	187325.260	360.865	316.792	23.514
CPSO	215994.852	361.742	333.274	24.701
SOMA	**152303.548**	369.904	**292.753**	**21.806**
OSOMA	154519.768	364.235	294.128	21.636
AS-SOMA_1	158684.477	359.309	295.200	21.849
A-MsPSO [48]	154433.900	**357.453**	295.209	21.829
AS-SOMA	**152235.210**	**355.851**	**292.412**	**21.805**

Table 7: LSTM optimization models with different algorithms run STLF problem independently for 10 times on RMSE.

	GA	PSO	CPSO	SOMA	OSOMA	AS-SOMA_1	A-MsPSO	AS-SOMA
Min	352.9138	358.0152	355.5573	355.2943	**350.7799**	351.3755	355.1319	353.197
Med	358.9311	360.9034	358.733	360.1304	358.4004	356.6651	356.0180	**355.8513**
Max	366.1678	363.6862	361.742	369.904	364.2347	359.3086	359.3801	**358.8697**
Mean	358.8283	360.8652	358.4011	360.5965	358.0656	356.7104	357.4532	**355.8722**
Std	3.837617	1.972727	2.030256	3.970404	3.424973	2.299549	3.0352617	**1.763565**
P	0.027918	1.14E-05	0.011311	0.0034	0.045681	0.19881	—	—
H	1	1	1	1	1	0	—	—

The bold values show the experimental results with good performance.

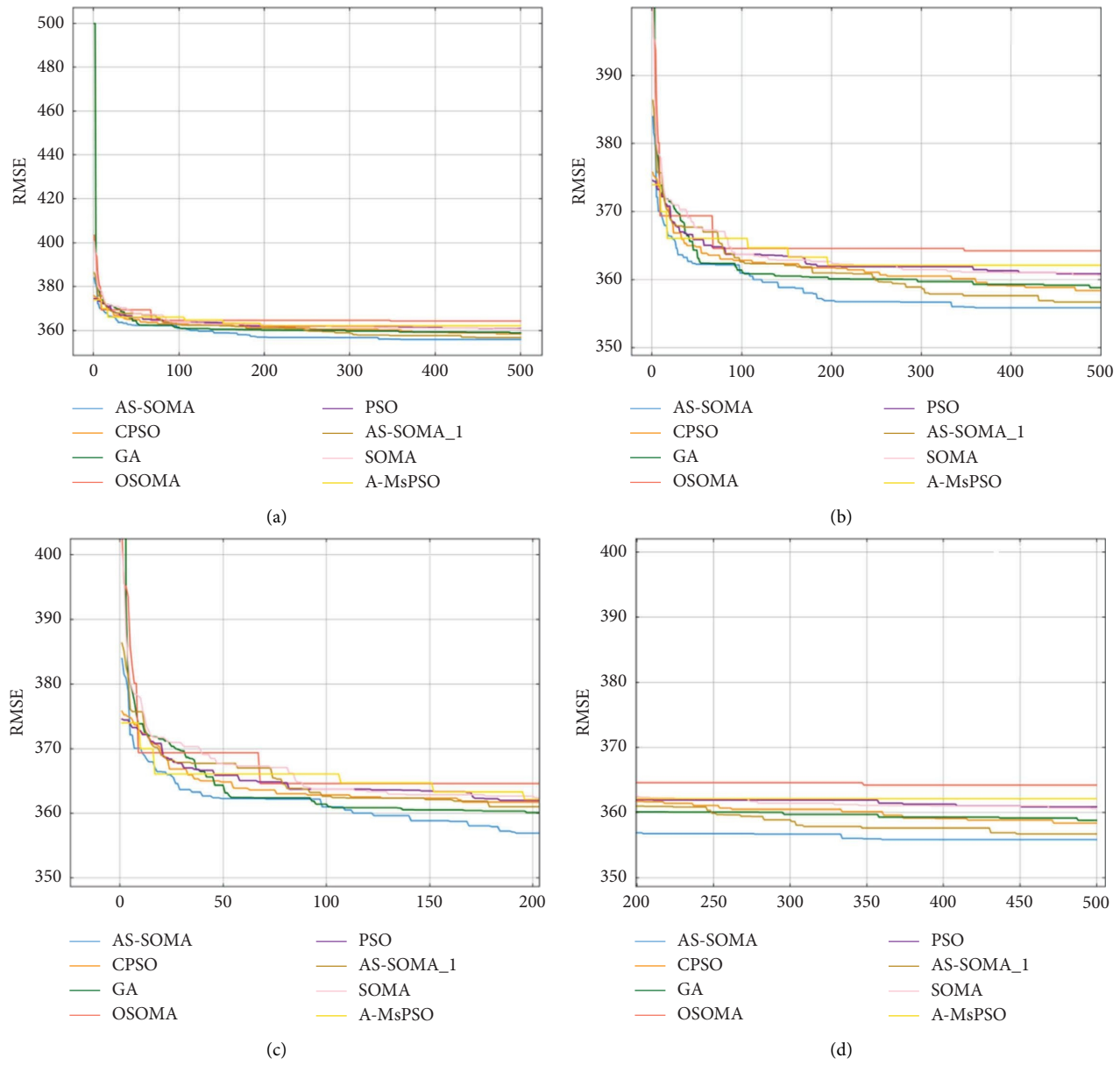


Figure 13: Average convergence curves of LSTM in STLF problem for 8 algorithms. (a) The algebraic convergence curve from 0 to 500. (b) The convergence curve trimmed on the y axis. (c) The algebraic convergence curve from 0 to 200. (d) The algebraic convergence curve from 200 to 500.

Figure 13(d) shows the situation of eight algorithms in the late search period. It can be seen that AS-SOMA has a fast convergence ability in the later stage of the search. The reason is that the improved scheme based on adaptive step size can solve the problem of poor convergence effectively, which is caused by the fixed step size. Additionally, the dynamical adjusting step size balances the development and the utilization of the algorithm.

In the early stage of the search, PSO is slightly ahead of SOMA, and SOMA gradually accelerates convergence ahead of PSO in the late stage of the search. The reason is that the movement of each particle in the PSO is guided by the optimal individual of the population. This guidance pushes the particles closer to $g$best in every dimension by subtracting the vector. Similarly, the movement of each particle is influenced by the leader of the population in SOMA. However, not all dimensions of particles migrate in all directions, and PRTVrctor interferes in the process of migration. Therefore, in the process of LSTM structure optimization, the full-dimensional migration of searching the early PSO can make the population approach better network parameters quickly. However, the multidimensional motion tends to make the population skip the optimal parameter when the population concentrates near the optimal solution in the late searching period. SOMA can immobilize certain dimensions and migrate others. It will approach the optimal solution gradually by controlling the direction.

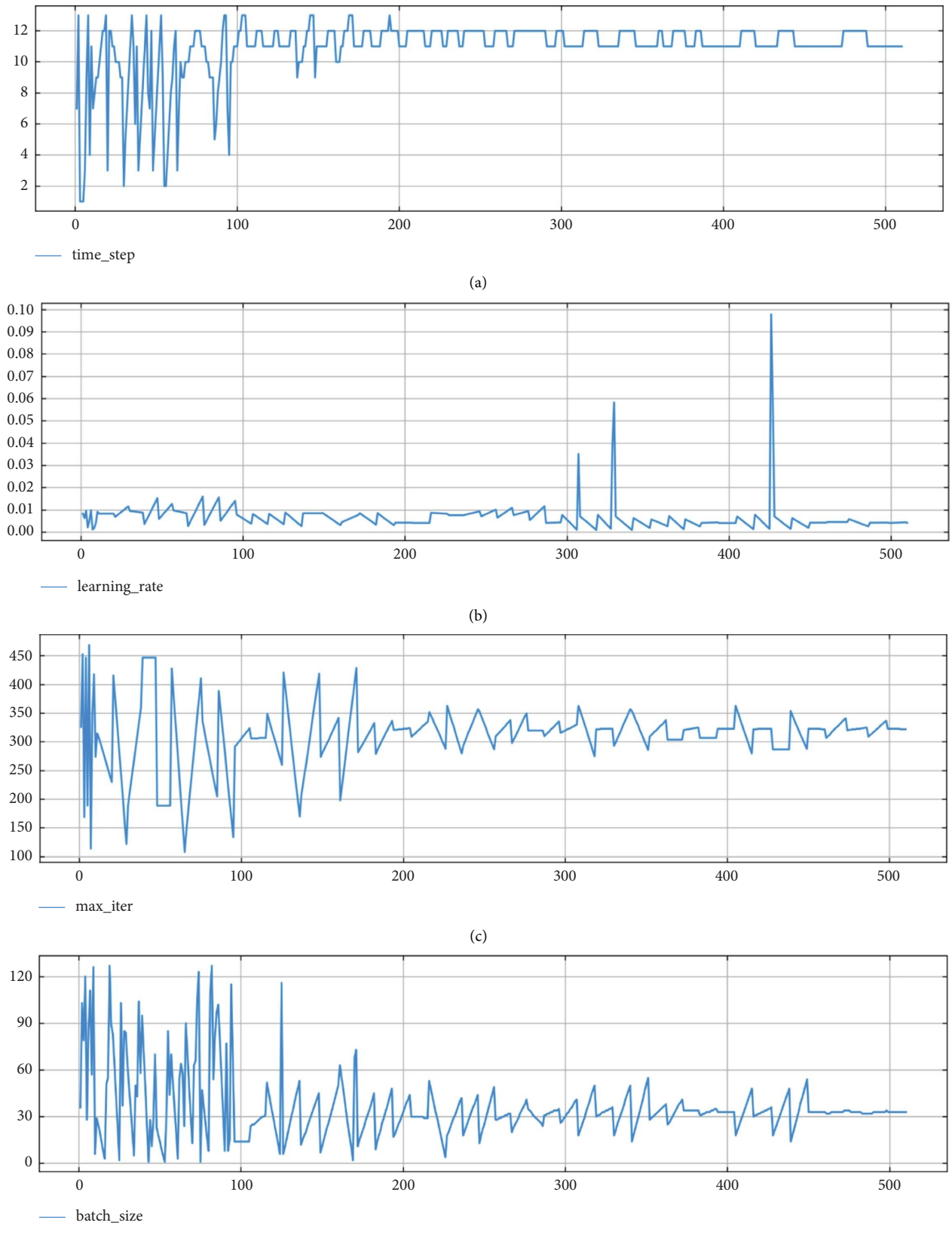


Figure 14: The changes of LSTM hyperparameters converge with AS-SOMA. (a) The change process of time step. (b) The change process of learning rate. (c) The change process of epochs. (d) The change process of batch size.

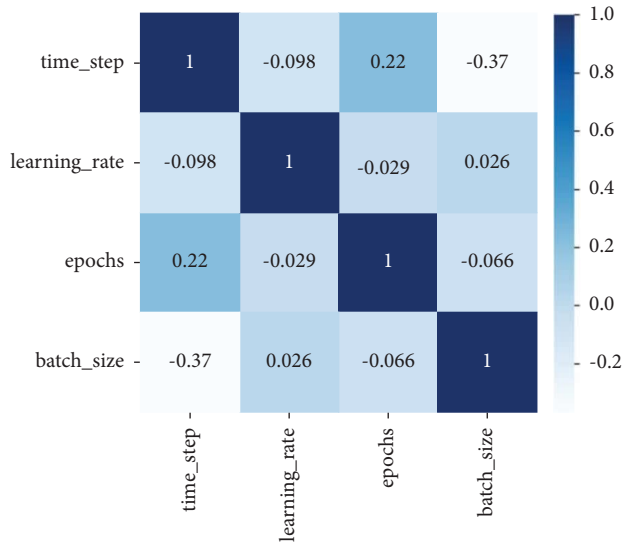


Figure 15: The correlation between LSTM hyperparameters in AS-SOMA convergence process.

The convergence process of AS-SOMA is verified in the LSTM hyperparameter variation in Figure 14. In the early stage of the search, the LSTM prediction model presents certain fluctuations in the time step, epochs, and batch size. The hyperparameters quickly stabilize after the 100th optimization and gradually tend to the optimal. Figure 14(b) shows that the learning rate presents a certain steady state in the process of AS-SOMA convergence.

Figure 15 shows the correlation degree among time step, learning rate, epochs, and batch size, whose absolute values are 0.37, 0.026, 0.066, and 0.029, respectively. It has the highest correlation between time step and batch size, with a value of 0.37. Therefore, it is necessary to analyse the hyperparameter constraint conditions in the process of combinatorial optimization in order to better train the network.

This paper constructs the AS-SOMA-LSTM optimization model to solve the STLF problem. The original SOMA is improved by a logistic chaotic and adaptive step size method, and its solution accuracy and convergence speed are verified. The LSTM optimization model based on AS-SOMA is proposed and applied to the practical problem of STLF. First, the optimal number of hidden layers is determined by the exhaustive method. Then, the remaining five hyperparameters are optimized by using the AS-SOMA, including the number of neurons, time step, learning rate, epochs, and batch size. Simulation experiments show that the AS-SOMA-LSTM model has significant advantages over other methods.

Data Availability

The dataset used to support the findings of this study can be accessed upon request.

Consent

Not applicable.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Chang Wang, Zijian Cao, and Xiaofeng Rong conceptualized the study; Xiaofeng Rong and Zijian Cao developed methodology; Chang Wang and Hanghang Zhou investigated the study; Hanghang Zhou, Chang Wang, and Linjuan Fan wrote the original draft of the manuscript; Xiaofeng Rong and Zijian Cao reviewed and edited the article and work to polish the whole study; the authors acquired fund. All authors have read and agreed to the published version of the manuscript.

Acknowledgments

This research was partially funded by the Shaanxi Natural Science Basic Research Project (2020JM-565) and the Science and Technology Plan Project Industry-University-Research Collaborative Innovation Plan of Weiyang District, Xi'an City, Shaanxi Province (K20200167).

References

[1] C. Lai, Y. Wang, K. Fan et al., "An improved forecasting model of short-term electric load of papermaking enterprises for production line optimization," *Energy*, vol. 245, Article ID 123225, 2022.

[2] T. Bashir, C. Haoyong, M. F. Tahir, and Z. Liqiang, "Short term electricity load forecasting using hybrid prophet-LSTM model optimized by BPNN," *Energy Reports*, vol. 8, pp. 1678–1686, 2022.

[3] D. Yang, J. e Guo, S. Sun, J. Han, and S. Wang, "An interval decomposition-ensemble approach with data-characteristic-driven reconstruction for short-term load forecasting," *Applied Energy*, vol. 306, Article ID 117992, 2022.

[4] S. N. Fallah, M. Ganjkhani, S. Shamshirband, and K. W. Chau, "Computational intelligence on short-term load forecasting: a methodological overview," *Energies*, vol. 12, no. 3, p. 393, 2019.

[5] G. Dudek, "Pattern similarity-based methods for short-term load forecasting Part 1: Principles," *Applied Soft Computing*, vol. 37, pp. 277–287, 2015.

[6] R. Chen, C. S. Lai, C. Zhong et al., "MultiCycleNet: multiple cycles self-boosted neural network for short-term electric household load forecasting," *Sustainable Cities and Society*, vol. 76, Article ID 103484, 2022.

[7] N. Andriopoulos, A. Magklaras, A. Birbas et al., "Short term electric load forecasting based on data transformation and statistical machine learning," *Applied Sciences*, vol. 11, no. 1, p. 158, 2020.

[8] J. F. Chen, W. M. Wang, and C. M. Huang, "Analysis of an adaptive time-series autoregressive moving-average (ARMA) model for short-term load forecasting," *Electric Power Systems Research*, vol. 34, no. 3, pp. 187–196, 1995.

[9] J. Huo, T. Shi, and J. Chang, "Comparison of Random Forest and SVM for electrical short-term load forecast with different data sources," in *Proceedings of the 2016 7th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, pp. 1077–1080, IEEE, Beijing, China, August, 2016.

[10] A. Lahouar and J. Ben Hadj Slama, "Day-ahead load forecast using random forest and expert input selection," *Energy Conversion and Management*, vol. 103, pp. 1040–1051, 2015.

[11] C. Cecati, J. Kolbusz, P. Różycki, P. Siano, and B. M. Wilamowski, "A novel RBF training algorithm for short-term electric load forecasting and comparative studies," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 10, pp. 6519–6529, 2015.

[12] J. Zhu, H. Dong, W. Zheng, S. Li, Y. Huang, and L. Xi, "Review and prospect of data-driven techniques for load forecasting in integrated energy systems," *Applied Energy*, vol. 321, Article ID 119269, 2022.

[13] M. Alhussein, K. Aurangzeb, and S. I. Haider, "Hybrid CNN-LSTM model for short-term individual household load forecasting," *IEEE Access*, vol. 8, pp. 180544–180557, 2020.

[14] X. Guo, Y. Gao, Y. Li, D. Zheng, and D. Shan, "Short-term household load forecasting based on Long- and Short-term Time-series network," *Energy Reports*, vol. 7, pp. 58–64, 2021.

[15] M. Aouad, H. Hajj, K. Shaban, R. A. Jabr, and W. El-Hajj, "A CNN-Sequence-to-Sequence network with attention for residential short-term load forecasting," *Electric Power Systems Research*, vol. 211, Article ID 108152, 2022.

[16] P. Bento, J. Pombo, and S. Mariano, "Short-term load forecasting using optimized LSTM networks via improved bat algorithm," in *Proceedings of the 2018 International conference on intelligent systems (IS)*, pp. 351–357, IEEE, Funchal, Portugal, September, 2018.

[17] H. Hu, X. Xia, Y. Luo, C. Zhang, M. S. Nazir, and T. Peng, "Development and application of an evolutionary deep learning framework of LSTM based on improved grasshopper optimization algorithm for short-term load forecasting," *Journal of Building Engineering*, vol. 57, Article ID 104975, 2022.

[18] W. Zhang and T. Wang, "Short-term power load forecasting model design based on EMD-PSO-GRU," *Scientific Programming*, vol. 2022, Article ID 4755519, 8 pages, 2022.

[19] N. Xu, J. Chang, and X. Nie, "AME: attention and memory enhancement in hyper-parameter optimi-zation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 480–489, New Orleans, LA, USA, June, 2022.

[20] J. S. Judd, "Learning in neural networks," in *Proceedings of the First Annual Workshop on Computational Learning Theory*, pp. 2–8, Cambridge, MA, USA, August, 1988.

[21] T. Li, J. Shi, W. Deng, and Z. Hu, "Pyramid particle swarm optimization with novel strategies of competition and cooperation," *Applied Soft Computing*, vol. 121, Article ID 108731, 2022.

[22] Y. Xu, C. Hu, Q. Wu et al., "Research on particle swarm optimization in LSTM neural networks for rainfall-runoff simulation," *Journal of Hydrology*, vol. 608, Article ID 127553, 2022.

[23] J. Wang, J. Cao, and S. Yuan, "Shear wave velocity prediction based on adaptive particle swarm optimization optimized recurrent neural network," *Journal of Petroleum Science and Engineering*, vol. 194, Article ID 107466, 2020.

[24] J. Ma, D. Xia, H. Guo et al., "Metaheuristic-based support vector regression for landslide displacement prediction: a comparative study," *Landslides*, vol. 19, no. 10, pp. 2489–2511, 2022.

[25] Y. Wang, H. Tang, J. Huang, T. Wen, J. Ma, and J. Zhang, "A comparative study of different machine learning methods for reservoir landslide displacement prediction," *Engineering Geology*, vol. 298, Article ID 106544, 2022.

[26] I. Zelinka and J. Lampinen, "SOMA–self-organizing migrating algorithm mendel," in *Proceedings of the 6th International Con-Ference on Soft Computing*, Brno, Czech Republic, September, 2000.

[27] Z. Cao, H. Jia, T. Zhao, Y. Fu, and Z. Wang, "An adaptive self-organizing migration algorithm for parameter optimization of wavelet transformation," *Mathematical Problems in Engineering*, vol. 2022, Article ID 6289215, 14 pages, 2022.

[28] L. Skanderova, T. Fabian, and I. Zelinka, "Self-organizing migrating algorithm using covariance matrix adaptation evolution strategy for dynamic constrained optimization," *Swarm and Evolutionary Computation*, vol. 65, Article ID 100936, 2021.

[29] P. R. Dhal, P. K. Pradhan, and M. K. Muni, "Improving navigational parameters during robot motion planning using SOMA technique," *Intelligent Systems*, pp. 179–188, Springer, Singapore, 2022.

[30] K. Deep, "A new hybrid self organizing migrating genetic algorithm for function optimization," in *Proceedings of the 2007 IEEE Congress on Evolutionary Computation*, pp. 2796–2803, IEEE, Singapore, September, 2007.

[31] S. Agrawal and D. Singh, "Modified Nelder-Mead self organizing migrating algorithm for function optimization and its application," *Applied Soft Computing*, vol. 51, pp. 341–350, 2017.

[32] W. He, T. Y. Lee, and J. van Baar, "DynamicsExplorer: visual analytics for robot control tasks involving dynamics and LSTM-based control policies," in *Proceedings of the 2020 IEEE Pacific Visualization Symposium (PacificVis)*, pp. 36–45, IEEE, Tianjin, China, June, 2020.

[33] A. Graves, N. Jaitly, and A. Mohamed, "Hybrid speech recognition with deep bidirectional LSTM," *IEEE*, in *Proceedings of the 2013 IEEE workshop on automatic speech recognition and understanding*, pp. 273–278, Olomouc, Czech Republic, December, 2013.

[34] Y. Li, Z. Zhu, D. Kong, H. Han, and Y. Zhao, "EA-LSTM: evolutionary attention-based LSTM for time series prediction," *Knowledge-Based Systems*, vol. 181, Article ID 104785, 2019.

[35] Y. Hua, Z. Zhao, R. Li, X. Chen, Z. Liu, and H. Zhang, "Deep learning with long short-term memory for time series prediction," *IEEE Communications Magazine*, vol. 57, no. 6, pp. 114–119, 2019.

[36] A. H. Bukhari, M. A. Z. Raja, M. Sulaiman, S. Islam, M. Shoaib, and P. Kumam, "Fractional neuro-sequential ARFIMA-LSTM for financial market forecasting," *IEEE Access*, vol. 8, pp. 71326–71338, 2020.

[37] S. Banik, N. Sharma, M. Mangla, and S. N. Mohanty, "LSTM based decision support system for swing trading in stock market," *Knowledge-Based Systems*, vol. 239, Article ID 107994, 2022.

[38] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, no. 5-6, pp. 602–610, 2005.

[39] I. Zelinka, "SOMA—self-organizing migrating algorithm," *Self-Organizing Migrating Algorithm*, pp. 3–49, Springer, Berlin, Germany, 2016.

[40] J. J. Liang, B. Y. Qu, and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2015 competition on learning-based real-parameter single objective optimization," vol. 29Technical Report 201411A, pp. 625–640, Computational Intelligence Laboratory, Zhengzhou University, Singapore, 2014.

[41] X. Tang, L. Zhuang, J. Cai, and C. Li, "Multi-fault classification based on support vector machine trained by chaos particle swarm optimization," *Knowledge-Based Systems*, vol. 23, no. 5, pp. 486–490, 2010.

[42] T. Kadavy, M. Pluhacek, and A. Viktorin, "Self-organizing migrating algorithm with clustering-aided migration and adaptive perturbation vector control," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp. 1916–1922, Lille, France, July, 2021.

[43] Z. Y. Lin and L. L. Wang, "Opposition-based self-organizing migrating algorithm," *Computer Science*, vol. 39, no. 5, 2012.

[44] F. U. M. Ullah, N. Khan, T. Hussain, M. Y. Lee, and S. W. Baik, "Diving deep into short-term electricity load forecasting: comparative analysis and a novel framework," *Mathematics*, vol. 9, no. 6, p. 611, 2021.

[45] M. Tan, S. Yuan, S. Li, Y. Su, H. Li, and F. H. He, "Ultra-short-term industrial power demand forecasting using LSTM based hybrid ensemble learning," *IEEE Transactions on Power Systems*, vol. 35, no. 4, pp. 2937–2948, 2020.

[46] J. Ponoćko and J. V. Milanović, "Forecasting demand flexibility of aggregated residential load using smart meter data," *IEEE Transactions on Power Systems*, vol. 33, no. 5, pp. 5446–5455, 2018.

[47] Z. Zhang, "Improved Adam optimizer for deep neural networks," in *Proceedings of the 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, pp. 1-2, IEEE, Banff, Alberta, Canada, June, 2018.

[48] S. Zdiri, J. Chrouta, and A. Zaafouri, "An expanded heterogeneous particle swarm optimization based on adaptive inertia weight," *Mathematical Problems in Engineering*, vol. 2021, Article ID 4194263, 2024 pages, 2021.