EAI Endorsed Transactions
on Energy Web
Research Article EAI.EU

Optimal scheduling of renewable power distribution systems combining deep learning and particle swarm optimization

Qiuyong Yang[1], Bin Chen[1],Yanlu Huang[2*], Xudong Hu[2]

[1]China Southern Power Grid Company Limited, Guangzhou, Guangdong, 528518, China
[2]Southern Power Grid Artificial Intelligence Technology Co., LTD, Guangzhou, Guangdong, 528518, China

Abstract:

Introduction: To address the optimal scheduling problem of renewable power distribution systems, this paper proposes an integrated framework combining deep learning-based load forecasting with particle swarm optimization.
Objectives: The first objective is accurate multivariate load forecasting (cooling, heating, and electric loads). The second objective is minimizing network loss by optimally scheduling electric vehicle (EV) charging locations and times.
Methods: For load forecasting, a hybrid RCNN-SVR model is constructed. The convolutional neural network (CNN) acts as a feature extractor to implicitly capture representative patterns from input data, while support vector regression (SVR) produces the final load predictions. Missing and outlier data are pre-processed. For optimal scheduling, a dual-layer particle swarm optimization (PSO) algorithm is developed. The inner layer enforces system constraints, and the outer layer minimizes network loss. EV charging load is simulated using the Monte Carlo method, and two cases (variable vs. fixed charging addresses) are optimized.
Results: Experimental results demonstrate that the proposed RCNN-SVR model achieves high prediction accuracy, with mean absolute percentage error as low as 2.41% for winter electric loads. The dual-layer PSO reduces peak system load from 11.2×10^3 kW to 10.5×10^3 kW and decreases network loss by 2.13%, effectively smoothing grid fluctuations.
Conclusion: The RCNN-SVR model significantly improves multivariate load prediction accuracy compared to separate forecasting methods. The dual-layer PSO successfully converts disorderly EV charging into orderly scheduling, reducing peak load and network loss. Together, they provide a practical solution for renewable distribution system scheduling.

Keywords: Renewable distribution system; Optimal dispatch; Deep learning; Convolutional neural network (CNN); Support vector regression (SVR); Load forecasting

Received on 06 April 2026, accepted on 13 May 2026, published on 01 June 2026

doi: 10.4108/ew.12491

*Corresponding author Email: YanluHuang_011@outlook.com

1. Introduction

Energy shortage and environmental degradation are becoming more and more serious, and new energy generation instead of non-renewable energy has become one of the important solutions to the problem. Unlike traditional power generation methods, new energy distributed power generation such as photovoltaic and wind power generation are generally close to the load side, with more flexible power supply methods, avoiding excessive transmission losses, and high efficiency and low investment costs. The use of electric vehicles can not only

reduce the consumption of non-renewable energy, but also reduce carbon emissions, which can help solve the deteriorating environmental problems. There are many studies on modeling the distributed PV output and the charging load power of electric vehicles. The study [1] proposed a dual closed-loop control strategy and constant power control strategy for three-phase voltage PWM converters during charging and discharging, and built a simulation model of V2G charging and discharging device in PSCAD/EMTDC. However, the analysis is complicated because the charging load is related to people's travel purpose and road location conditions. The study [2] concluded that the randomness of light and wind speed can cause errors in the prediction of PV and wind turbine output, and showed that the current prediction errors of PV and wind turbine generally satisfy the normal distribution, and simulated three PV and wind turbine output curves. The study [3] uses the Monte Carlo simulation method, which selects the charging time of electric vehicles and the random number of charge states to calculate the charging load, and the results calculated by this method are relatively accurate but not efficient. Electric vehicle charging load and distributed power supply also have an impact on the distribution network, and this impact can be mainly divided into steady-state impact and transient impact [4]. The study [5] analyzed the impact of electric vehicles on the distribution network mainly in terms of the pervasiveness of electric vehicles. The five factors are the degree of penetration, type, charging method, charging time and charging characteristics. The study [6] conducted a study on the impact of distributed PV and EV access to the distribution network, respectively. The study [7] analyzed the impact of EV load on the planning, power quality and economic aspects of operation of the distribution network.

Load forecasting is an important basic work to ensure the safe, stable and economic operation of the energy system, which is based on historical (sample) data, summarizing the variation law between load and time, parametrizing and modeling the law, and making projections of future load changes through modeling algorithms based on the "inertia principle" such as time series or trend extrapolation, or data-driven learning algorithms [8]. The algorithm can be used to predict future load changes through time series or trend extrapolation, or data-driven learning algorithms [8]. With the increasing research and application of artificial intelligence methods, a large number of load forecasting studies have been carried out by deep learning models such as neural networks [9]. The integrated energy system (IES), as an important component of the new generation energy system, takes the distribution system as the core and integrates various forms of energy supply, energy conversion and energy storage devices in the system, thus realizing the coupling of different types of energy sources, networks and loads [10][11] [12]. However, due to the large volume of historical load data of integrated energy systems and their high complexity and stochasticity, it is generally difficult to achieve accurate and reliable results in practical applications by following the single model forecasting approach for power systems [13]. Take the support vector machine method [14] as an example, although it has achieved excellent results in power load forecasting as an emerging intelligent algorithm, it is difficult to guarantee the data quality requirements for integrated energy systems with a large variety of data types and data volumes, and in addition, the implementation of the model procedure is complicated and it is not possible to determine whether the knowledge in the data is redundant [15]. The load forecasting model with a single forecast quantity is no longer suitable for the current complex integrated energy distribution system, and researchers have combined various models based on the strengths and weaknesses of various algorithms to improve the forecasting accuracy [16]. In recent years, the application of deep learning methods in the field of load forecasting has received much attention from researchers both at home and abroad in the context of upgrading computational tools and increasing the amount of training data on a large scale, and these methods have better robustness in dealing with various external influences than traditional load forecasting methods [17-18]. Therefore, it is necessary to introduce deep learning algorithms into the load forecasting process of integrated energy systems based on load correlation, taking into account the seasonal, meteorological, and date related influences [19], to reasonably pre-process the data and improve the data quality, so as to effectively improve the prediction accuracy of the model, reflect the coupling between different types of energy sources, and assist the economic safety of the whole system. This can effectively improve the prediction accuracy of the model, reflect the coupling between different types of energy sources, and assist in the economic and safe operation of the whole system.

The proposed renewable distribution system differs from existing distributed energy frameworks in three key aspects. Structurally, it integrates high-penetration renewables and EV loads without requiring islanding capability or centralized aggregation. Operationally, it employs a joint RCNN-SVR and dual-layer PSO framework that simultaneously performs multivariate load forecasting (cooling, heating, electric) and constraint-aware EV scheduling — a capability not present in conventional microgrids or VPPs. This unique combination advances adaptive power architectures by converting disorderly EV charging into orderly, loss-minimizing behavior while capturing cross-energy coupling effects, thereby improving system efficiency and reducing peak-to-valley load differences.

In this paper, we consider the optimal scheduling problem of renewable power distribution system from two directions of deep learning and particle swarm optimization simultaneously. The main contributions of this paper are summarized as follows: (1) We use a dual particle swarm algorithm to find the optimal solution for the optimal scheduling of the location and time of charging of electric vehicles connected to the distribution network, using an electric vehicle example. (2) A multivariate load forecasting method based on deep learning architecture is proposed. (3) The feasibility and validity of the model are

verified by using actual data of an integrated energy system.

2. Related Work

2.1 Regression Convolutional Neural Network Model

A convolutional neural network (CNN) is a type of feedforward neural network with convolutional computation. layer, activation layer, fully connected layer and output layer. The convolutional layer and the pooling layer are usually multiple, and are often used interchangeably. The name of convolutional neural network is related to the computational mechanism of the convolutional layer, in which each neuron in the output eigenface is locally connected to its input, and the corresponding connection weights are weighted and summed with the local input plus a bias value to obtain the input value of the neuron, which is the same principle as the convolution process, hence the name. CNNs are composed of multiple independent neurons and are suitable for large-scale data learning problems, with features such as local feature acquisition, weight region sharing, and quadratic sampling.

The structure of a simple CNN network used as a regression problem for processing samples, i.e., a regression convolution neural network (RCNN) model, is given in Figure 1. In this model, the input layer is the influencing factor, and the last regression layer outputs the load value. When used for integrated energy system prediction, the dimensionality of the feature information is reduced after several convolutional and pooling layers are stacked during the feature extraction process. This information carries the relevant data features that more clearly reflect the load changes, thus reducing the impact of the uncertainty of the influencing factors on the model prediction results. Combining this model with other models can help improve the overall model learning efficiency and prediction accuracy.

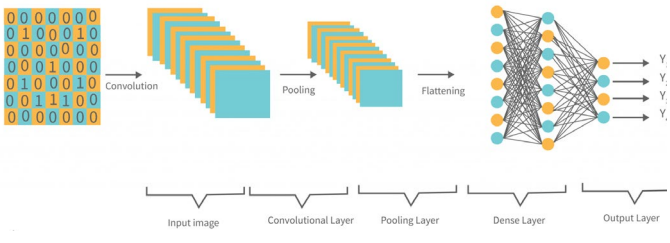


Figure 1. Structure of RCNN model

2.2 Support vector regression model

The core of the integrated energy system load forecasting model is to train the load data and build the load forecasting model. The regression model based on support vector machine, which is commonly used in load forecasting, is introduced. The output of the support vector regression (SVR) model is a continuous number, not a category of output for solving classification problems, and there is a tolerance margin in SVR. However, this model has the same goal as the basic support vector machine model in solving the classification problem: to minimize the error and maximize the classification margin. Figure 2 depicts the 1-dimensional SVR model, where and denote the axes of the feature space coordinate system consisting of the input data and the learning target, denotes the loss function, and denotes the tolerance margin of the corresponding input data. The model aims to optimize the following functions.

$$\begin{cases} \min_{\omega,b,\xi_l,\xi_l^*} \left(\frac{1}{2}\|\omega\|^2 + \lambda\sum_l(\xi_l + \xi_l^*)\right) \\ \text{s.t.} \begin{cases} y_l - (\omega^{\mathrm{T}}\mathbf{X}_l + b) \le \varepsilon + \xi_l \\ y_l - (\omega^{\mathrm{T}}\mathbf{X}_l + b) \le \varepsilon + \xi_l^* \\ \xi_l, \xi_l^* \ge 0, \forall l \end{cases} \end{cases} \quad (1)$$

Where: $\boldsymbol{X}_l$ is the input data set; y_l is the corresponding class tag of the lth data; ω is an n -dimensional vector; b is a real bias; λ is a coefficient controlling the width of the classification margin; and ξ_l, ξ_l^* are non-negative relaxation variables. The processing of the sample data in the SVR structure is shown in Figure 3.

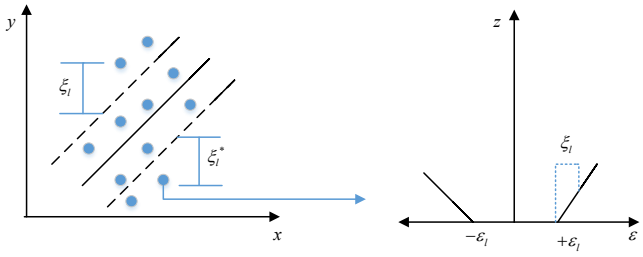


Figure 2. Dimensional Linear SVR

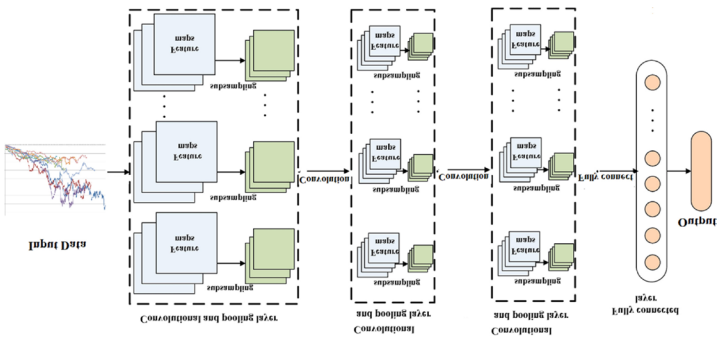


Figure 3. SVR model prediction structure diagram

2.3 Particle swarm algorithm

Particle swarm optimization algorithm is a process in which the optimal particles in a population of particle followers keep approaching to the optimal solution in the solution space. That is, the particle swarm optimization algorithm requires a certain amount of initial solutions, and then iterates to find the optimal solution. In the process of each iteration, the particles need to be updated by two quantities, namely, the individual's historical optimal solution, which is its own optimal since so many iterations, and the population's historical optimal solution, which is the optimal solution found by all particles since many iterations. Now, suppose that the population consists of N particles in a D -dimensional solution space, where the position of the ith particle can be expressed as a row vector $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD}), i = 1,2,\ldots,N$. The "velocity" of the ith particle can also be expressed as a D -dimensional The "velocity" of the ith particle can also be expressed as a D -dimensional row vector $V_i = (v_{i1}, v_{i2}, \ldots, v_{iD}), i = 1,2,\ldots,N$. The optimal position searched by the ith particle until the latest iteration is called the individual optimal position, which can be written as $p_{best} = (p_{i1}, p_{i2}, \ldots, p_{iD}), i = 1,2,\ldots,N$. The optimal position found by this particle swarm until the latest iteration is called the population optimal position, which can be written as $g_{best} = (p_{g1}, p_{g2}, \ldots, p_{gD})$. According to the above definition, the whole particle population can be described as $\{X_{(k)1}, X_{(k)2}, \ldots, X_{(k)N}\}$. k represents the number of iterations where the particles are located. After finding the individual optimal position and the population optimal position, the particles update their velocity and position vectors according to equations (2) and (3)

$$V_i^{(k+1)} = \omega \cdot V_i^{(k)} + c_1 \cdot r_1 \cdot \left(p_{\text{best}}^{(k)} - X_i^{(k)}\right) + c_2 \cdot r_2 \cdot \left(g_{\text{best}}^{(k)} - X_i^{(k)}\right) \quad (2)$$

$$\boldsymbol{X}_i^{(k+1)} = \boldsymbol{X}_i^{(k)} + \boldsymbol{V}_i^{(k+1)} \quad (3)$$

Where ω is called the inertia factor and usually takes the value of 1. c_1, c_2 are called the learning factors and often called the acceleration factors. r_1, r_2 are random numbers belonging to 0 to 1 and obeying uniform distribution. The flow chart of the basic particle swarm algorithm is shown in Figure 4.

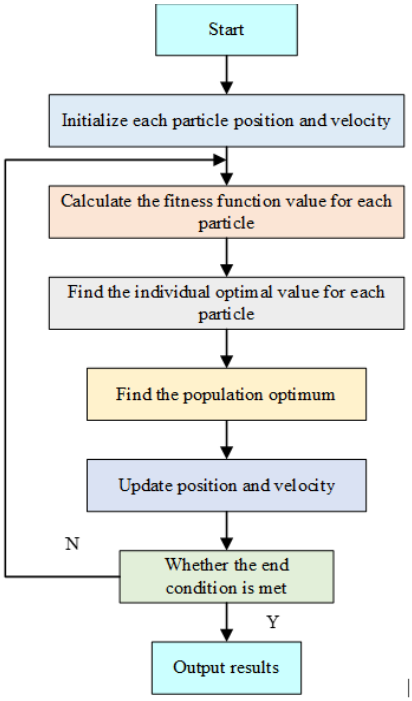


Figure 4. Flow chart of particle swarm optimization algorithm

3. Methodology

3.1 RCNN-SVR model

In order to address the accuracy shortcomings of the support vector machine model in multivariate load forecasting for integrated energy systems due to data problems, this paper combines a deep learning-based convolutional neural network with a support vector regression model and designs an RCNN-SVR model. Specifically, the model uses more representative high-level features extracted from the RCNN to train the SVR classifier, and then uses the trained SVR classifier to predict the load values. The RCNN-SVR architecture is shown in Figure 5. The RCNN-SVR model combines the advantages of both approaches, allowing more implicit features to be extracted and using less computation time to train the model. There are two main steps in the model: feature extraction by RCNN model and prediction by SVR model. In addition, the RCNN network is fine-tuned to make feature extraction more efficient by adding another convolutional layer and a maximum pooling layer. To reduce errors and prevent overpopulation, a dropout strategy is used, i.e., a random deactivation layer is added after the maximum pooling layer 3 with a scale of 0.3. The placement of the dropout layer after the pooling stage is designed to regularize high-level feature representations, where overfitting is more likely to occur. A dropout rate of 0.3 is selected based on empirical evaluation to provide a balance between retaining sufficient information and preventing excessive co-adaptation of neurons. This

configuration effectively reduces overfitting while maintaining stable learning, thereby improving the generalization capability of the RCNN-SVR model.

This dropout strategy plays a critical role in improving the generalization ability of the model by randomly deactivating neurons during training, thereby preventing overfitting to specific data patterns. It also enhances the stability and robustness of the RCNN-SVR model by ensuring that the learned feature representations are less sensitive to noise and variations in the input data. As a result, the model achieves more reliable performance when applied to unseen data. For each of the three convolutional layers, the filter size is 1 × 1 and the number of filters is 20, 25, and 50, respectively. The design of the RCNN architecture is guided by the need to effectively capture temporal and nonlinear patterns in multivariate load data. The convolutional layers are configured to extract local feature representations from input sequences, enabling the model to identify underlying relationships among different load variables. The use of pooling layers reduces dimensionality and helps in retaining dominant features while minimizing noise sensitivity. This combination of convolution and pooling operations supports hierarchical feature extraction, allowing the model to learn both low-level and high-level representations. These design choices enhance the model's ability to generalize and improve forecasting accuracy in complex load prediction scenarios. The selection of convolutional layer configurations and filter dimensions is guided by both theoretical considerations and empirical performance evaluation. The use of 1×1 convolutional filters enable efficient channel-wise feature transformation while reducing computational complexity, allowing the model to capture essential relationships among input variables without increasing parameter size. The progressive increase in the number of filters (20, 25, and 50) facilitates hierarchical feature extraction, enabling the network to learn more abstract and representative patterns at deeper layers. In addition, the combination of convolutional and pooling layers enhances the robustness of feature extraction by reducing noise sensitivity and improving generalization. These design choices contribute to improved prediction accuracy and stability of the RCNN-SVR model. Convolution and pooling layers reduce feature dimensionality by removing redundant information while preserving key patterns. This lowers computational cost, improves training efficiency, and enhances prediction accuracy and generalization.

In addition, the last two layers (fully connected and regression layers) are removed because the combination of SVR models eliminates the need to extract other implied features from these two layers, and the feature size of the last random deactivation layer is the same as the feature size in the maximum pooling layer 2 of the RCNN. The RCNN-SVR model is applied to predict the multivariate load of an integrated energy distribution system by first extracting the features of the input factors and predicting the load using the trained SVR classifier. In the training phase, the implied features extracted from the RCNN model are used to train the SVR classifier, and the predicted load values are compared with the real values, and the convergence of the model is checked. Before being input into the SVR model, the features extracted from the RCNN are preprocessed and normalized to ensure consistent scaling and to improve convergence during training. These processed features are then mapped into a higher-dimensional feature space using an appropriate kernel function, enabling the SVR model to capture nonlinear relationships between input variables and load values. This structured pipeline, combining feature extraction and kernel-based regression, enhances the overall prediction accuracy and robustness of the model. The performance of the SVR model is influenced by key parameters, including the regularization parameter C and the tolerance margin ϵ\epsilonϵ. The parameter C controls the trade-off between minimizing training error and maintaining model generalization, where larger values emphasize accuracy while smaller values improve robustness. The tolerance margin ϵ\epsilonϵ defines the allowable error range within which no penalty is assigned, enabling the model to ignore minor deviations and reduce overfitting. Appropriate selection of these parameters ensures stable regression performance and improved prediction accuracy within the combined RCNN-SVR framework.

The integration of support vector regression with RCNN-extracted features is motivated by the need to improve regression performance for complex and heterogeneous load patterns. The RCNN model transforms raw input data into high-level feature representations that capture nonlinear relationships and interactions among variables. These transformed features provide a more informative and structured input space for the SVR model, enabling it to perform more accurate regression compared to using raw data directly. In particular, the SVR model effectively handles nonlinear mappings through kernel functions, which enhances prediction accuracy across different load categories such as electric, heating, and cooling loads. This combination improves the overall robustness and generalization capability of the forecasting framework. If the convergence condition is not met, the training phase is executed again. The training process is controlled through defined convergence and iteration criteria to ensure model stability and reproducibility. The model is trained iteratively over multiple epochs, and convergence is determined based on the stabilization of prediction error between successive iterations. Specifically, the training process is terminated when the reduction in error falls below a predefined threshold or when the maximum number of iterations is reached.

The loss evaluation is performed using standard regression error metrics, such as mean squared error (MSE) during training and mean absolute percentage error (MAPE) for performance assessment. These metrics enable continuous monitoring of model performance and ensure that the trained model achieves stable and reliable prediction accuracy.

In the test phase, input factors from the test dataset are used and load values are predicted at this time. For integrated energy systems with complex input factors, the proposed combined model relies on more information contained in the implicit features extracted from the RCNN model to provide the SVR model with enough information for training, thus solving the problem of lack of accuracy due to the high data requirements of the support vector machine model and establishing a better relationship between the predicted and actual load values.

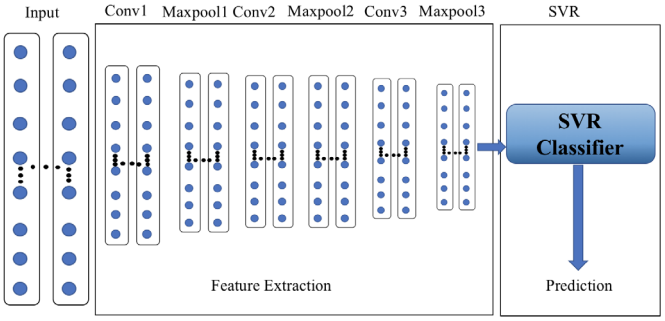


Figure 5. RCNN-SVR model

A comprehensive data preprocessing pipeline is applied to improve data quality and model robustness. Missing values are handled using interpolation techniques, while outliers are detected and removed based on statistical thresholds to prevent distortion of training patterns. Furthermore, all input features are normalized to ensure consistent scaling and stable model convergence. Feature selection is performed based on correlation analysis to retain the most relevant variables, thereby reducing redundancy and enhancing prediction performance. These preprocessing steps contribute to improved reliability and reproducibility of the forecasting results. The selection of input influencing factors is performed using Pearson correlation and t-distribution testing. Features with strong correlation (e.g., r>0.5r) are retained, while weakly correlated ones are removed. A significance level (e.g., p<0.05) ensures only statistically reliable features are selected. These thresholds improve prediction accuracy by reducing irrelevant inputs and enhancing model robustness.

3.2 Dual particle swarm algorithm

To improve analytical clarity, the optimal dispatch problem is formulated as follows. The objective is to minimize the total network loss of the distribution system:

$$\min F = \sum_{t=1}^{T} P_{\text{loss}}(t) \quad (4)$$

where $P_{\text{loss}}(t)$ represents the network loss at time interval t, and T is the total number of time periods.

Recents

Subject to the following constraints:
(1) Power balance constraint:

$$P_{\text{gen}}(t) + P_{\text{PV}}(t) = P_{\text{load}}(t) + P_{\text{EV}}(t) + P_{\text{loss}}(t) \quad (5)$$

(2) EV charging power constraint:

$$0 \le P_{\text{EV},i}(t) \le P_{\text{EV},i}^{\max} \quad (6)$$

(3) Node voltage constraint:

$$V_i^{\min} \le V_i(t) \le V_i^{\max} \quad (7)$$

(4) Line capacity constraint:

$$S_{ij}(t) \le S_{ij}^{\max} \quad (8)$$

where P_{gen} , P_{PV}, P_{load} , and P_{EV} denote generated power, photovoltaic power, load demand, and EV charging load, respectively. V_i is the voltage at node i, and S_{ij} represents the apparent power flow between nodes.

The optimal scheduling problem of electric vehicles in this paper is actually a discrete optimization problem, and because of the constraints, there is a high probability that the position and velocity vectors in the particle swarm optimization algorithm will not satisfy the constraints after the particles are updated. The second layer of the particle swarm optimization algorithm is nested in the original swarm optimization algorithm to solve the problem that the updated positions and velocities do not satisfy the constraints. According to the role of the algorithm, the first layer of the two-layer swarm optimization algorithm is called the outer layer and the second layer is called the inner layer: The design of the dual particle swarm optimization algorithm is motivated by the need to efficiently handle constraint satisfaction in discrete scheduling problems. The outer layer is responsible for exploring the global search space and optimizing the primary objective, while the inner layer focuses on enforcing system constraints by refining the particle positions generated by the outer layer. This interaction allows the outer layer to maintain optimization efficiency, while the inner layer ensures feasibility of solutions. By separating objective optimization and constraint handling into two coordinated layers, the algorithm achieves a balance between search performance and constraint satisfaction, leading to more reliable and stable scheduling results.
(1) Initialize the outer swarm, including the number of particles, the position vector and the velocity vector of each particle that meets the constraints.
(2) Calculate the fitness function of the outer swarm for each particle based on the position vector of each outer particle.

(3) For each outer particle, compare its fitness function value with the individual optimal fitness function value, and if it is smaller than the current individual optimal fitness value, replace the current individual optimal fitness value with that fitness function value, otherwise, keep it unchanged.

(4) For each outer particle, compare its fitness function value with the population optimal fitness function value of the outer particle, and if it is smaller than the current population optimal fitness value, replace the current population optimal fitness value with the fitness function value, otherwise, keep it unchanged.

(5) Update the position vector and velocity vector of each outer particle.

(6) Initialize the inner particle population, including the number of particles, the position vector and the velocity vector of each particle. At the same time, the individual optima used are substituted by the individual optima of the outer particles, and the population optima of the inner particles are substituted by the population optima of the outer particles.

(7) The position vector of each inner particle is used to calculate the fitness function of each particle corresponding to the inner particle population.

(8) For each inner particle, compare its fitness function value with the individual optimal fitness function value, if it is smaller than the current individual optimal fitness value, then replace the current individual optimal fitness value with the fitness function value, otherwise, keep it unchanged. For each inner particle, compare its fitness function value with the population optimal fitness function value of the inner particle, and if it is smaller than the current population optimal fitness value, replace the current population optimal fitness value with the fitness function value, otherwise, keep it unchanged.

(9) Update the position vector and velocity vector of each inner layer particle. If the inner fitness function has reached the constraint of the model or has reached the predetermined number of iterations of the inner particle population, exit the iterative update cycle of the inner particle population. If the constraint is satisfied, the optimal position and velocity vectors obtained by the inner swarm optimization algorithm are used as the result of the outer swarm particle update. If the upper limit of iterations is reached and the inner swarm cycle is exited, the position and velocity of the outer swarm particles remain the same as before the update.

(10) If the predetermined number of iterations of the outer swarm has been reached or the predetermined error range has been reached, the outer iteration of the update is withdrawn and the solution of the problem is obtained; otherwise, it returns to process (2).

According to the above analysis of the two-layer swarm optimization algorithm and the calculation process of the two-layer swarm algorithm, it can be seen that the fitness function of the outer swarm of the two-layer swarm algorithm is related to the problem itself, while the fitness function of the inner swarm is the embodiment of the constraints of the problem, and the role of the inner swarm algorithm is to make the position and velocity vectors obtained by the outer swarm update meet the requirements of the constraints, so as to ensure the reasonableness of the final optimization results and the compliance with the constraints. The function of the inner swarm algorithm is to make the position and velocity vectors obtained from the outer swarm meet the constraints and ensure the rationality of the final optimization result.

4. Experiments

4.1 Optimized scheduling based on deep learning

This paper implements multivariate load analysis and prediction of integrated energy distribution system based on MATLAB platform. The sample includes three different types of load data, i.e., electric load, cooling/heating load, and hot water supply load. In order to prove the year-round adaptability of the model, the load data and influencing factors from July 1 to July 31, 2017 were used as the original training sample data for summer prediction, and the corresponding information of typical working days and typical rest days in summer were selected as the test sample data for prediction analysis. The load data and influencing factors from January 1 to January 31, 2017 were used as the original training sample data for winter prediction, and the corresponding information of typical working days and typical rest days in winter was selected as the test sample data for prediction analysis. In addition to the load data, the park system also provides other raw data, including data on temperature, humidity, wind speed, radiation and other influencing factors in hourly steps for each time of the system, as well as information on the relevant date types, based on the adequate collection devices in the park center. The pre-processing of the historical load data and the influencing factors is described in Section 2.4. The correlation analysis is performed on the relevant data, and the factors with correlation coefficients >0.3 are selected as inputs for training the model after t-distribution test based on the Pearson coefficient values. By selecting the input factors, the uncertainty in predicting and measuring the influencing factors can be reduced. The Pearson correlation coefficients are shown in Table 1 for a typical winter month.

Table 1. Pearson correlation coefficient between electrical load and influencing factors

Type of correlation factor	Absolute value of coefficient

First 1h electric load	0.9448
First 1h cooling/heating load	0.3918
First 1h hot water supply load	0.3068
First 2h electric load	0.8136
First 2h cooling/heating load	0.2627
First 3h electric load	0.6685
Temperature	0.1382
Radiation	0.1615
Wind speed	0.1778
Hour	0.0358
Week Type	0.2283
Holiday	0.3332

After the training of the model, the actual load data was selected as the test set for load prediction. Taking the typical rest days in winter and summer as an example, the multivariate load prediction results are shown in Figure 6 and Figure 7 after inputting the influencing factors in hourly steps into the model. From Figure 6 and Figure 7, it can be seen that the trends of the prediction curves in summer and winter are close to the actual curves, which proves that the model is adaptable throughout the year and has high accuracy. The average absolute percentage error, the average precision and the average precision of the weights are used to measure the accuracy of the load prediction results, which are shown in equations (6), (7) and (8), and the results are shown in Table 2. The results are shown in Table 2. From Table 2, it can be seen that the prediction results considering multivariate load correlations have good prediction accuracy. When the three types of loads, namely, electric load, cooling/heating load, and hot water load, are applied to the prediction by considering meteorological factors and social factors, the prediction results of this model can be better. The feasibility and validity of the proposed model is verified by comparing the average prediction results of the same category of loads with the support vector machine load prediction model for integrated energy systems considering the effects of natural gas loads and meteorological factors in the literature, as shown in Table 3. The results are shown in Table 3. The results are shown in Table 4 when comparing the indexes of this method with the indexes of electric, cooling and thermal load forecasting separately.

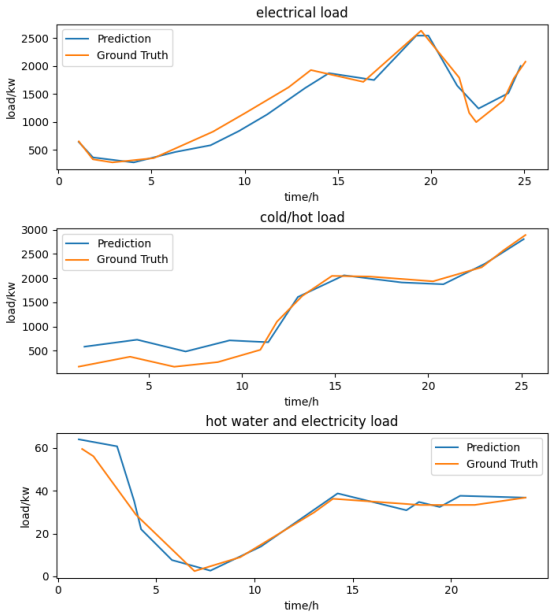


Figure 6. Comparison of winter multivariate load forecast results

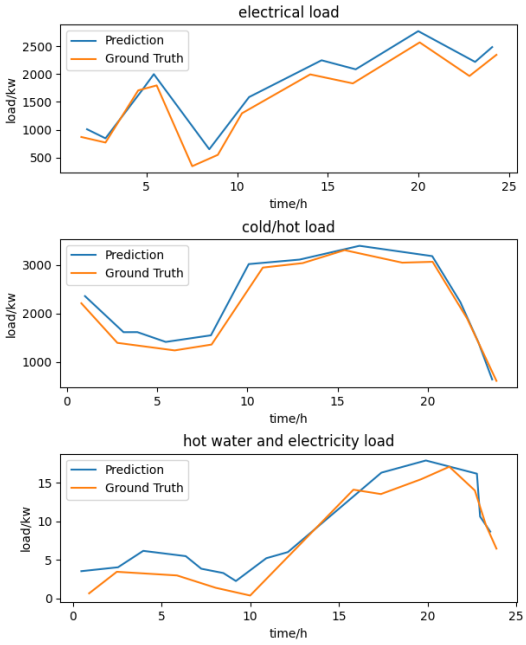


Figure 7. Comparison of multivariate load forecast results in summer

Table 2. Model evaluation indexes

Load Type	Average absolute percentage error/%	Average accuracy/%	Average accuracy of weights/%
Summer weekday electric load	3.69	97.36	96.42
Summer weekday cooling/heating load	3.26	96.75	96.42
Summer weekday hot water supply load	1.43	95.88	96.42
Summer Holiday Electric Load	4.56	95.56	96.17
Summer rest day cooling/heating load	2.67	97.36	96.17
Hot water supply load on summer rest days	2.90	97.08	96.17
Winter weekday electric load	2.41	97.58	97.32
Winter weekday cooling/heating load	2.63	97.36	97.32
Winter weekday hot water load	4.70	95.44	97.32
Winter Rest Day Electric Load	2.73	97.28	96.77
Winter rest day cooling/heating load	4.06	95.95	96.77
Hot water supply load on winter rest day	3.68	96.33	96.77

Table 3. Comparison of model results

Models	Model of this paper	Literature Model
Average absolute percentage error/%	3.26	7.55

Table 4. Comparison of mean absolute percentage errors

Load Type	Average absolute percentage error/%	
	Methodology of this paper	Separate prediction of electric cooling and heating
Summer weekday electric load	3.69	18.35
Summer weekday cooling/heating load	3.26	14.67
Summer weekday hot water supply load	4.13	19.14
Summer Holiday Electric Load	4.56	19.59
Summer rest day cooling/heating load	2.67	12.69
Hot water supply load on summer rest days	2.90	15.25
Winter weekday electric load	2.41	16.43
Winter weekday cooling/heating load	2.63	13.83
Winter weekday hot water load	4.70	19.22
Winter Rest Day Electric Load	2.73	12.55
Winter rest day cooling/heating load	4.06	10.87
Hot water supply load on winter rest day	3.68	12.79

Table 4 shows that, compared with the methods for predicting electric, cooling, and thermal loads separately,

the prediction method that takes into account load correlation and influence factors can reduce the average absolute error of electric, cooling, and thermal loads and improve the accuracy of the model. In conclusion, the multivariate load prediction method based on regression convolutional neural network and support vector regression proposed in this paper is feasible and efficient, and the prediction accuracy of hot and cold electric loads can be effectively improved by fully exploiting the correlation between multivariate loads.

4.2 Monte Carlo Simulation for EV Charging Load

The stochastic nature of EV charging behaviour is modelled using Monte Carlo simulation with the following assumptions: each EV charges once per day at a constant power of 3 kW, continuing until 90% state of charge (SOC), with initial SOC following a uniform distribution $\mathcal{U}(0.20,0.50)$, and no vehicle-to-grid discharge is considered. Daily arrival time (home) follows a truncated normal distribution with mean 18:00 and standard deviation 120 minutes bounded between 16:00-22:00, while daily departure time follows a truncated normal with mean 08:00 and standard deviation 90 minutes bounded between 06:00-10:00, based on National Household Travel Survey data. Daily driving distance follows a log-normal distribution ($\mu_{\ln} = 3.8, \sigma_{\ln} = 0.8$), reflecting a right-skewed pattern where most trips are short and few are long. The simulation uses $M = 10{,}000$ independent iterations with simple random sampling and a fixed random seed (42) to ensure reproducibility, with convergence verified by monitoring the running mean of total charging load, which stabilizes within $\pm 1\%$ after approximately 5,000 iterations. The simulated aggregate load profile is validated against empirical data from the Pecan Street project, with a KolmogorovSmirnov test confirming no significant difference between simulated and empirical arrival time distributions ($p > 0.05$). The expected charging load at each time step is then computed by averaging across all iterations and EVs, producing a realistic uncertain load profile for input to the dual-layer PSO optimization. Monte Carlo simulation is used to model EV charging load by treating variables such as arrival time, departure time, and state of charge as random inputs. These are generated using probability distributions and sampled over multiple iterations to ensure reliable and realistic representation of uncertainty.

4.3 Dual Particle Swarm Optimization Scheduling

In this paper, the MATLAB software is used to simulate the charging location optimization model proposed above. In this model, it is assumed that the charging power distribution of all EVs connected to the example system is fixed during the charging time, and what needs to be optimized is the node location of each EV connected to the example system. The objective function of the charging location optimization model is the total network loss of the example system, and the constraints are as shown in the above section.

Before starting the simulation, it is necessary to set the simulation parameters, including the parameters of the electric vehicle and the photovoltaic power supply, as well as the parameters of the standard particle swarm optimization algorithm. (1) Electric vehicle parameters setting. The number of EVs connected to the system is 800, and the power of each EV is 3 kW when charging. The travel start time, travel end time and charging time of EVs are all integers. (2) The parameters of the photovoltaic power supply are set. The photovoltaic power supply is connected to the example system from node 6 and node 7, and the power connected to the two nodes is 50% of the photovoltaic power output. (3) Standard particle swarm optimization algorithm parameter setting. The number of particles is set to 80, the maximum number of iterations is 20, and the learning factors c1 and c2 are set to 1.5.

The performance of the particle swarm optimization algorithm is closely related to the selection of its control parameters, particularly the inertia weight and the cognitive and social coefficients. In this study, fixed learning factors c1=c2=1.5are selected to maintain a balanced influence between individual and global search behaviors. This choice ensures that particles neither converge prematurely nor diverge excessively.

The inertia weight plays a key role in controlling the exploration–exploitation trade-off. A relatively moderate inertia weight is considered to allow sufficient global search capability in early iterations while enabling stable convergence in later stages. Although a constant parameter setting is adopted in this work, the convergence behavior observed in Figures 8 and 11 indicates that the selected parameters provide stable and efficient optimization performance, with fast convergence and reduced oscillations.

These parameter settings contribute to achieving a balance between exploration and exploitation, thereby improving optimization stability and ensuring reliable scheduling results under the given problem constraints.

After running the program, the optimization results of the fitness function value, i.e., the network loss value, are obtained (as shown in Figure 8). From Figure 8, it can be seen that after 15 iterations, the network loss value is reduced from the initial 548.5 kW.h to 545.8 kW.h. After the 15th iteration, the standard particle swarm optimization starts to converge and the optimization amount of the network loss value is 1.7%. The optimization results show that the standard particle swarm optimization algorithm converges faster, takes less time to operate, and has a better optimization effect, which is suitable for solving optimization problems with relatively simple constraints. The position vector of the population optimal particles obtained after the standard particle swarm optimization algorithm can be used to obtain the node position of each

electric vehicle connected to the example system. The initial random access (shown in Figure 10) is also shown.

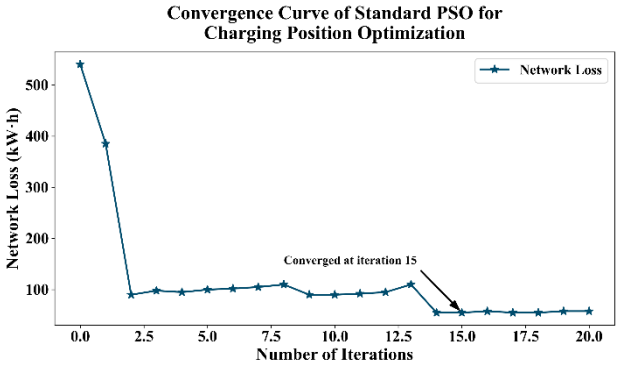


Figure 8. Optimization of charging position

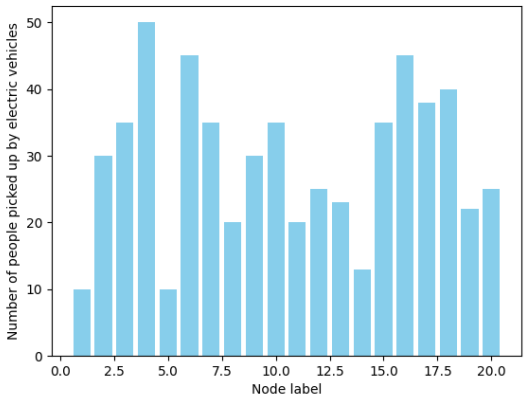


Figure 9. Electric vehicle access after optimization

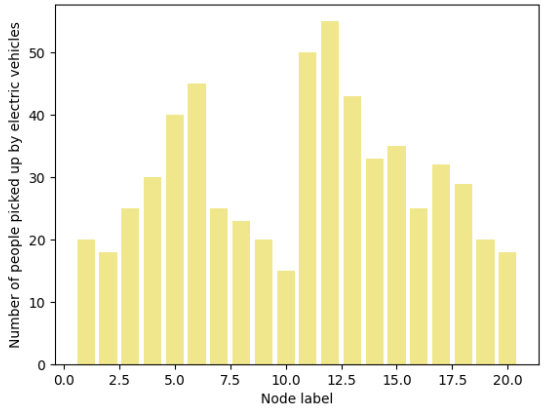


Figure 10. Electric vehicle access before optimization

The initial distribution of EV access to the example system is relatively even, with a small difference in the amount of access to each node, but after optimization by the standard particle swarm optimization algorithm, it can be seen from Figure 9 that the EVs are more distributed in the nodes from node 1 to node 10 and from node 22 to node 29 compared to the other nodes in the example system. This is because nodes 1 to 10 and 22 to 29 are basically the nodes near the input of each branch, i.e., these nodes are closer to the power node 0. When the load is close to the power node in the system, the power loss of the whole system will be relatively reduced. Therefore, the initial EVs are connected to the system as shown in Figure 9 after optimization. This simulation is an optimization of the node locations in the distribution system for EV charging, and is a solution of the charging location scheduling strategy. Therefore, the charging power distribution habits of EV owners during charging hours are not changed, but the charging access location is simply scheduled, and the charging behavior of owners is not affected. In fact, the simulation results are mainly for the power supply side, and the power supply related departments should make reasonable planning when establishing the new intelligent power distribution network and reasonably allocate the charging facilities to the users of electric vehicles, so as to reduce the losses in the operation of the intelligent power distribution system and let the system operate safely, efficiently and economically. The users are reasonably allocated charging facilities to reduce the losses in the operation of the intelligent power distribution system, so that the system can operate safely, efficiently and economically. Since the charging power distribution of EV owners has not changed during the rechargeable hours, EVs are still charged in a disorderly manner, so there is still much room for optimizing the network loss of the system. The following simulation will continue to optimize the charging time of EVs, so that EVs can be charged in an orderly manner.

Due to the complex structure of the two-layer swarm and the constraints of the charging time optimization model, the data calculation is relatively large, so the running time of the program is also relatively long, which takes 2h 7min.

The computational performance of the dual particle swarm optimization algorithm depends on the number of particles, iterations, and problem dimensionality. The time complexity can be approximated as O(N×T×D), where N is the number of particles, T is the number of iterations, and D represents the dimensionality of the optimization problem. Due to the dual-layer structure, the inner optimization loop increases computational cost and memory usage compared to standard PSO.

However, this additional computation improves constraint satisfaction and solution quality. As the number of electric vehicles and system nodes increases, the algorithm exhibits scalable behavior, with computation time growing proportionally with problem size. The iteration process remains stable without significant oscillations, indicating efficient convergence. Overall, the method provides a balanced trade-off between computational cost and optimization accuracy, making it suitable for practical large-scale scheduling problems.

The optimization results of the obtained network loss values are shown in Figure 11. After eight iterations, the net loss value is reduced from 550.72 KW.h to 539 KW.h.

After the ninth iteration, the two-layer particle swarm optimization starts to converge, and the total optimization of the net loss value is 2.13%. From the obtained optimization results, the bilayer swarm convergence is faster and the optimization effect is more obvious, which is suitable for solving the optimization model with complex constraints, but the computation time increases significantly with the increase of data volume. After optimizing the charging periods of electric vehicles by the two-layer particle swarm optimization algorithm, the optimal solution of the charging power distribution of 800 electric vehicles and the total load curve of the distribution system for 24 periods are obtained, and compared with the total load curve of the system before optimization when charging is disordered, as shown in Figure 12.

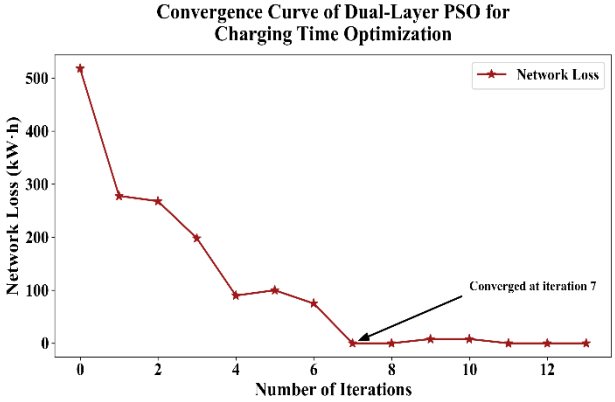
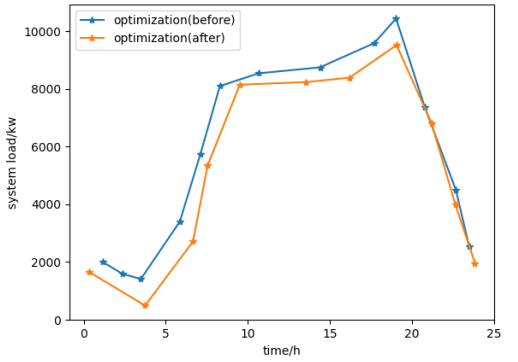
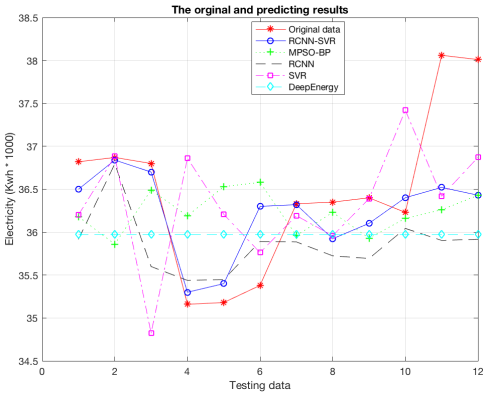


Figure 11. Electricity time optimization results graph





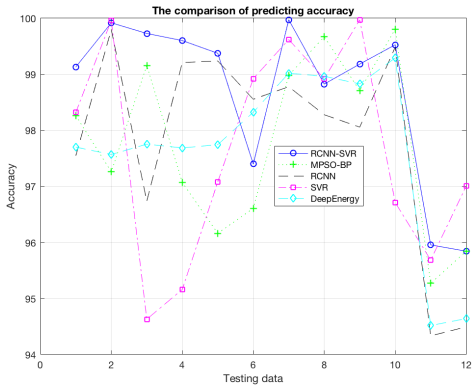


Figure 12. System load curve before and after charging time optimization

As can be seen from Figure 12, before optimization, the 800 electric vehicles connected to the example are charged in a disorderly manner, so that when superimposed on the basic load curve given in Figure 12, the maximum value of the original curve increases, resulting in a "peak on peak" effect Figure 13. The maximum value of the basic load curve is around 10.3×103 kW at 19 o'clock, and the maximum value of the system load curve increases to 11.2×103 kW after adding electric vehicles and photovoltaic power sources, Figure 14 which increases the peak-to-valley difference of the load curve and makes the load fluctuation bigger. After the double-layer particle swarm optimization algorithm optimizes the charging time of electric vehicles, Figure 15 the maximum value of the system load is reduced to about 10.5×103 kW, which significantly reduces the peak load of the system. At the same time, the trough load value of the system increases, because some of the peak load of the system is transferred to the trough load, which reduces the peak-to-trough difference of the load curve and the load fluctuation. It can be seen that the optimization of the charging period reasonably adjusts the charging power distribution of electric vehicles, and the disorderly charging becomes orderly charging, which can reduce the peak and fill the valley of the load curve and make the curve smoother with the connection of photovoltaic power Figure 16.

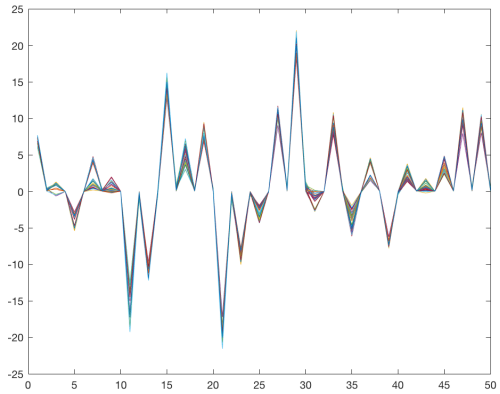


Figure 13. computation based on simulated data

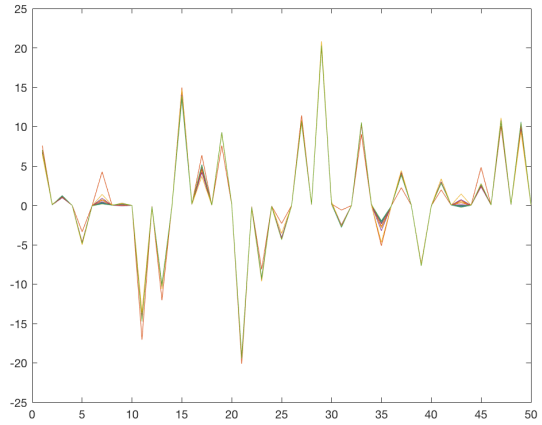


Figure 14. calculation based on analyzed data

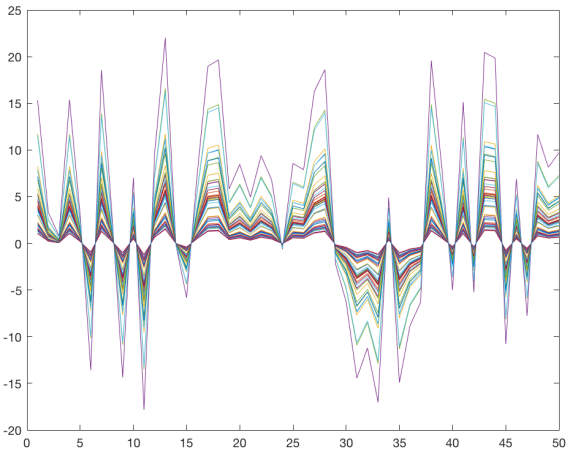


Figure 15. Multiple data series showing synchronized fluctuations

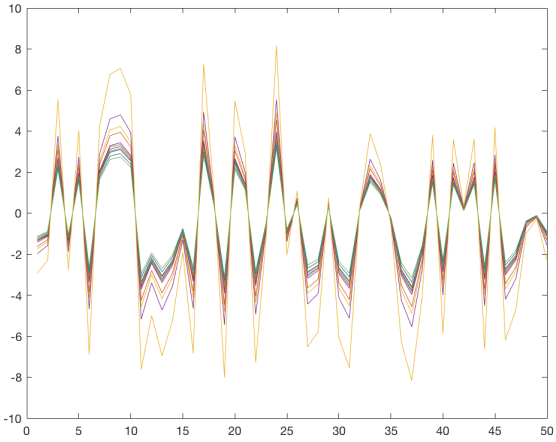


Figure 16: Multi-series plot showing fluctuating trends

5. Conclusion

This paper presents an integrated framework combining deep learning and particle swarm optimization for optimal scheduling of renewable power distribution systems. The RCNN-SVR model effectively improves multivariate load forecasting accuracy by extracting representative features and reducing prediction errors. Experimental results show that the model achieves a mean absolute percentage error as low as 2.41%, significantly outperforming traditional methods.

In addition, the dual-layer particle swarm optimization algorithm enhances scheduling performance by effectively handling system constraints and minimizing network loss. The optimization results demonstrate a reduction in peak load from 11.2×10^3 kW to 10.5×10^3 kW and a decrease in network loss by 2.13%, while also smoothing load fluctuations.

Overall, the proposed approach successfully integrates forecasting and optimization, providing a practical and efficient solution for improving the operational performance and reliability of renewable power distribution systems.

Future research can extend the proposed Monte Carlo-based EV charging simulation by incorporating real-time and dynamic operating conditions, such as stochastic user behaviour, traffic patterns, and renewable energy variability. Additionally, integrating real-time data streams and adaptive control strategies can enhance the responsiveness of the scheduling framework in practical smart grid environments. Further improvements may include large-scale implementation, distributed optimization, and the integration of advanced forecasting models to better handle uncertainty and improve system scalability.

While this paper focuses on minimizing network loss as the single objective, real-world distribution system operation involves multiple competing objectives including operational cost, voltage stability, and reliability. A multi-objective optimization would consider trade-offs among these criteria. For example, minimizing network loss may shift EV charging to off-peak hours, which reduces loss but may concentrate charging during low-voltage periods, potentially affecting voltage stability. Similarly, minimizing cost under time-of-use pricing may conflict with loss minimization if price signals do not align with loss patterns. Comparative evaluation suggests that network loss minimization aligns reasonably well with voltage stability (both improve with reduced line currents) but may partially conflict with reliability (loss-minimizing schedules may overload specific feeders). Under varying scenarios such as high EV penetration (>1,600 EVs) or weak grid conditions, multi-objective optimization becomes necessary. Future work will extend the proposed framework using multi-objective particle swarm optimization (MOPSO) to generate Pareto fronts, enabling

system operators to select optimal trade-offs based on real-time priorities.

For real-world deployment, the proposed integrated energy system framework must account for three categories of practical constraints. First, user behavior patterns include EV owner preferences for charging time (e.g., willingness to delay charging), tolerance for demand response signals, and heterogeneity in daily travel patterns. The current Monte Carlo simulation assumes homogeneous behavior; future work will incorporate survey-based behavioral models. Second, regulatory policies such as time-of-use electricity pricing, carbon emission caps, renewable portfolio standards, and grid connection codes directly impact optimal scheduling outcomes. The framework can accommodate these by adjusting cost coefficients in the objective function or adding policy constraints. Third, dynamic operating conditions including real-time price fluctuations, weather-dependent PV output variability, and contingency events (e.g., line outages) require adaptive scheduling. While the current approach uses day-ahead deterministic optimization, the RCNN-SVR load forecasting model is inherently adaptable to new data, enabling receding-horizon (model predictive control) implementation

Declarations

Funding

Funding: Technology Project of China Southern Power Grid Company Limited
Number: （ZBKJXM20220025）

Conflicts of Interest

The authors declared that they have no conflicts of interest regarding this work.

Data Availability

The experimental data used to support the findings of this study are available from the corresponding author upon request.

Authors' Contributions:

Qiuyong Yang: Conceptualization, methodology, supervision, and project administration.
Bin Chen: Data curation, software implementation, and validation.
Yanlu Huang: Investigation, formal analysis, writing – original draft, and corresponding author responsibilities.
Xudong Hu: Visualization, resources, and writing – review & editing.
All authors have read and approved the final manuscript.

References

[1] Chen K, Ning X, Xin X, Shi F, Zhang Q, Li C. Day-ahead optimal scheduling of an integrated energy system based on a piecewise self-adaptive particle swarm optimization algorithm[J]. Energies. 2022; 15(3): 690.

[2] Li Y, Wang R, Yang Z. Optimal scheduling of isolated microgrids using automated reinforcement learning-based multi-period forecasting[J]. IEEE Transactions on Sustainable Energy. 2021; 13(1): 159–169.

[3] Bian Z, Zhang Q. Combined compromise solution and blockchain-based structure for optimal scheduling of renewable-based microgrids: Stochastic information approach[J]. Sustainable Cities and Society. 2022; 76: 103441.

[4] Menos-Aikateriniadis C, Lamprinos I, Georgilakis PS. Particle swarm optimization in residential demand-side management: A review on scheduling and control algorithms for demand response provision[J]. Energies. 2022; 15(6): 2211.

[5] Cao Z, Han X, Lyons W, O'Rourke F. Energy management optimisation using a combined long short-term memory recurrent neural network–particle swarm optimisation model[J]. Journal of Cleaner Production. 2021; 326: 129246.

[6] Du J, Zhang Z, Li M, Guo J, Zhu K. Optimal scheduling of integrated energy system based on improved grey wolf optimization algorithm[J]. Scientific Reports. 2022; 12(1): 1–19.

[7] Rezaee Jordehi A. Enhanced leader particle swarm optimisation (ELPSO): A new algorithm for optimal scheduling of home appliances in demand response programs[J]. Artificial Intelligence Review. 2020; 53(3): 2043–2073.

[8] Chen H, Gao L, Zhang Z. Multi-objective optimal scheduling of a microgrid with uncertainties of renewable power generation considering user satisfaction[J]. International Journal of Electrical Power & Energy Systems. 2021; 131: 107142.

[9] Kim I, Kim B, Sidorov D. Machine learning for energy systems optimization[J]. Energies. 2022; 15(11): 4116.

[10] Pang X, Zhang X, Liu W, Li H, Wang Y. Optimal scheduling of cogeneration system with heat storage device based on artificial bee colony algorithm[J]. Electronics. 2022; 11(11): 1725.

[11] Zhang C, Xie T, Yang K, Ma H, Xie Y, Xu Y, et al. Positioning optimisation based on particle quality prediction in wireless sensor networks[J]. IET Networks. 2019; 8(2): 107–113.

[12] Li M, Yang S, Zhang M. Power supply system scheduling and clean energy application based on adaptive chaotic particle swarm optimization[J]. Alexandria Engineering Journal. 2022; 61(3): 2074–2087.

[13] Mei Y, Li B, Wang H, Wang X, Negnevitsky M. Multi-objective optimal scheduling of microgrid with electric vehicles[J]. Energy Reports. 2022; 8: 4512–4524.

[14] Hannan MA, Abdolrasol MG, Mohamed R, Al-Shetwi AQ, Ker PJ, Begum RA, et al. ANN-based binary backtracking search algorithm for VPP optimal scheduling and cost-effective evaluation[J]. IEEE Transactions on Industry Applications. 2021; 57(6): 5603–5613.

[15] Rohiem NH, Soeprijanto A, Putra DFU, Syai'in M, Sulistiawati IB, Zahoor M, et al. Resolving economic dispatch with uncertainty effect in microgrids using hybrid incremental particle swarm optimization and deep learning method[J]. Proceedings of the Pakistan Academy of Sciences: A Physical and Computational Sciences. 2021; 58(S): 119–129.

[16] Mianaei PK, Aliahmadi M, Faghri S, Ensaf M, Ghasemi A, Abdoos AA. Chance-constrained programming for optimal scheduling of combined cooling, heating, and power-based microgrid coupled with flexible technologies[J]. Sustainable Cities and Society. 2022; 77: 103502.

[17] Ju X, Liu X, Liu S, Xiao Y. Optimal scheduling of wind–photovoltaic power-generation system based on a copula-based conditional value-at-risk model[J]. Clean Energy. 2022; 6(4): 550–556.

[18] Ji H, Wang H, Yang J, Feng J, Yang Y, Okoye MO. Optimal schedule of solid electric thermal storage considering consumer behavior characteristics in combined electricity and heat networks[J]. Energy. 2021; 234: 121237.

[19] Wang F, Liao X, Fang N, Jiang Z. Optimal scheduling of regional combined heat and power system based on improved MFO algorithm[J]. Energies. 2022; 15(9): 3410.