World Electric Vehicle Journal

MDPI

Article

Electric Vehicle Charging Load Forecasting Method Based on Improved Long Short-Term Memory Model with Particle Swarm Optimization

Xiaomeng Yang [1], Lidong Zhang [1,*] and Xiangyun Han [2]

[1] School of Transportation and Logistics Engineering, Shandong Jiaotong University, Jinan 250023, China
[2] Department of Traffic Management and Engineering, Chongqing Police College, Chongqing 401331, China; letianhan@163.com
* Correspondence: zhanglidong@sdjtu.edu.cn; Tel.: +86-13791038708

Abstract: With the rapid global proliferation of electric vehicles (EVs), their integration as a significant load component within power systems increasingly influences the stable operation and planning of electrical grids. However, the high uncertainty and randomness inherent in EV users' charging behaviors render accurate load forecasting a challenging task. In this context, the present study proposes a Particle Swarm Optimization (PSO)-enhanced Long Short-Term Memory (LSTM) network forecasting model. By combining the global search capability of the PSO algorithm with the advantages of LSTM networks in time-series modeling, a PSO-LSTM hybrid framework optimized for seasonal variations is developed. The results confirm that the PSO-LSTM model effectively captures seasonal load variations, providing a high-precision, adaptive solution for dynamic grid scheduling and charging infrastructure planning. This model supports the optimization of power resource allocation and the enhancement of energy storage efficiency. Specifically, during winter, the Mean Absolute Error (MAE) is 3.896, a reduction of 6.57% compared to the LSTM model and 10.13% compared to the Gated Recurrent Unit (GRU) model. During the winter–spring transition, the MAE is 3.806, which is 6.03% lower than that of the LSTM model and 12.81% lower than that of the GRU model. In the spring, the MAE is 3.910, showing a 2.71% improvement over the LSTM model and a 7.32% reduction compared to the GRU model.

Keywords: electric vehicles (EVs); load forecasting; long short-term memory network; particle swarm optimization; deep learning

Academic Editor: Grzegorz Sierpiński

Received: 11 February 2025
Revised: 26 February 2025
Accepted: 3 March 2025
Published: 5 March 2025

Citation: Yang, X.; Zhang, L.; Han, X. Electric Vehicle Charging Load Forecasting Method Based on Improved Long Short-Term Memory Model with Particle Swarm Optimization. *World Electr. Veh. J.* **2025**, *16*, 150. https://doi.org/10.3390/wevj16030150

1. Introduction

Amid the escalating global energy crisis and increasing environmental pollution concerns, electric vehicles (EVs) have rapidly emerged as a sustainable transportation solution. However, the widespread adoption of EVs has posed new challenges for power grids. The charging load of EVs is inherently random and volatile, potentially disrupting the stable operation of power grids. Therefore, accurately forecasting fluctuations in EV load [1] is essential for effective grid planning and operation.

Traditional forecasting techniques encompass various methods, including Monte Carlo simulation and Kalman filtering. Reference [2] developed a Monte Carlo-based model that incorporates EV types, using the predicted number of EVs in a region as a basis while considering other factors as model parameters. Reference [3] proposed a hybrid algorithm combining time-series analysis and Kalman filtering, which improved

accuracy and facilitated the derivation of state and observation equations, effectively underscoring the benefits of hybrid algorithms in improving short-term load forecasting for power systems.

Electric load forecasting is inherently nonlinear and influenced by multiple factors. Machine learning, with its robust nonlinear mapping capabilities, has demonstrated efficacy in addressing nonlinear problems in load forecasting. Traditional machine learning techniques, such as support vector machines, decision trees, and random forests, are generally suited for smaller datasets and are effective in addressing nonlinear problems. Reference [4] addressed issues of low accuracy and inadequate consideration of seasonality in traditional EV load forecasting by proposing a seasonal EV charging load prediction model based on random forests. Reference [5] introduced an adaptive improvement method for Particle Swarm Optimization to solve high-dimensional EV charging load prediction models.

Deep learning-based forecasting methods utilize neural networks as parameter structures for optimization. Neural networks, also known as artificial neural networks, are mathematical models inspired by the way biological neural networks process and transmit information. Commonly used neural networks for power load forecasting include backpropagation (BP) networks, convolutional neural networks (CNNs), recurrent neural networks (RNNs), and emerging Transformer models. For example, Reference [6] proposed a spatiotemporal graph convolutional network (GCN+LSTM) that integrates graph convolutional networks (GCNs) with Long Short-Term Memory (LSTM) networks to improve the accuracy of electric vehicle (EV) charging demand predictions and alleviate traffic congestion in high-demand areas. Reference [7] integrated Kolmogorov-Arnold Networks (KANs) into traditional machine learning frameworks, specifically Convolutional Neural Networks (CNNs). Reference [8] developed a short-term forecasting method based on a bidirectional LSTM (BiLSTM) neural network optimized by a Sparrow Search Algorithm (SSA) and variational mode decomposition (VMD).

Moreover, the accuracy of EV charging load forecasting is significantly influenced by external factors, among which weather conditions play an essential role. Weather variables, such as temperature, humidity, wind speed, and solar radiation, have a significant impact on EV charging demand. Therefore, incorporating weather factors is crucial for enhancing the accuracy and reliability of forecasting models. Reference [9] proposed an EV charging load forecasting method that considers multiple influencing factors, including weather. Reference [10] explored the effects of meteorological conditions on the spatiotemporal distribution of EV charging loads at highway service areas, presenting a weather-inclusive forecasting model for these areas. Reference [11] proposed a hybrid forecasting framework that integrates Partial Least Squares Regression (PLSR) with a lightweight Gradient Boosting Machine (LightGBM), combined with Bayesian hyperparameter optimization and a second-order cone optimal scheduling scheme. This approach significantly improves short-term load forecasting accuracy, with the Mean Absolute Error (MAE) reduced by 12.7%. Reference [12] addressed the high randomness and spatial heterogeneity of electric vehicle charging load by proposing a machine learning forecasting framework based on a dual perspective of industrial parks and charging stations. By combining the MLP and LSTM algorithms, the framework achieves high-precision forecasting for weekdays (LSTM R^2 = 0.9283), holidays (R^2 = 0.9154), and weekends (MLP R^2 = 0.9586).

As the world's largest automobile market and leading producer of electric vehicles (EVs), China is experiencing rapid growth in the number of EVs in use. In China, there are significant regional differences in electric vehicle usage patterns and charging demands. Taking first-tier cities such as Beijing, Shanghai, and Shenzhen as examples, due to their dense populations and developed public transportation systems, electric vehicles are primarily used for urban commuting, and charging demand is concentrated during the

morning and evening peak hours on weekdays, as well as at residential charging stations at night. In some second- and third-tier cities, the use of electric vehicles is more diverse, including operational vehicles such as taxis and ride-hailing services, resulting in more scattered and uncertain charging demand.

In order to accurately predict the electric vehicle charging load for rational grid capacity planning, optimized charging facility layout, and maintaining power system stability, this study proposes a PSO-LSTM model that combines Particle Swarm Optimization (PSO) with the LSTM network to improve the accuracy of EV charging load prediction. Comparative experimental case studies between the PSO-LSTM model and traditional models demonstrate that the PSO-LSTM model significantly outperforms others in prediction accuracy, providing effective data support for grid and power system operations. This research not only validates the correctness and effectiveness of the PSO-LSTM model but also offers new perspectives and solutions for the field of electric vehicle charging load forecasting.

2. Model Description

2.1. Parameters and Data

To ensure the scientific rigor and reproducibility of this study, this research systematically set and rigorously validated key parameters during the construction and optimization of the electric vehicle charging load prediction model. These parameters cover core aspects such as data preprocessing, model architecture design, and optimization algorithm configuration. Table 1 provides a complete list of the definitions and settings of all parameters, allowing readers to fully understand the implementation details and key design logic of this study.

Table 1. Key parameter definitions in the study.

Parameter	Definition	Unit
D	The search space dimension of the PSO algorithm	
N	The number of particles in the PSO algorithm	
ω	The inertia weight of the PSO algorithm	
num	The number of iterations for the PSO algorithm	
c_1	The acceleration coefficient for personal best position in PSO	
c_2	The acceleration coefficient for the global best position in PSO	
r_1, r_2	Random numbers within the range [0, 1]	
P	Total energy consumption of EV charging stations	kWh
Hidden Units1	The number of neurons in the first hidden layer	
Hidden Units2	The number of neurons in the second hidden layer	
Dropout	The dropout rate	
Batch Size	Batch size	
Epochs	The number of training iterations	

To ensure the academic rigor of the expressions and consistency in reader comprehension, this study provides systematic definitions and standardized explanations of the technical term abbreviations used. Table 2 provides a complete list of all abbreviations used in the paper, along with their corresponding full forms, definitions, and application contexts, in order to eliminate potential ambiguities.

Table 2. Abbreviations and their full forms cited in the study.

Abbreviation	Full Form
PSO	Particle Swarm Optimization
LSTM	Long Short-Term Memory
PSO-LSTM	Particle Swarm Optimization Long Short-Term Memory
GRU	Gated Recurrent Unit
EV	Electric vehicle
BP	Backpropagation
CNN	Convolutional neural network
KAN	Kolmogorov-Arnold network
RNN	Recurrent neural network
GCN	Graph convolutional network
GNN	Graph neural networks
BiLSTM	Bidirectional LSTM
SSA	Sparrow Search Algorithm
VMD	Variational mode decomposition
MAE	Mean Absolute Error
MSE	Mean Squared Error
RMSE	Root Mean Squared Error

Traffic flow studies have shown that travelers' travel patterns exhibit distinct regularities, which, in turn, determine the patterns of urban road traffic conditions. As a result, the traffic flow fluctuations generated by travelers on the same road segment also follow a regular pattern, including periodic trends with yearly, quarterly, monthly, weekly, and daily cycles. Collecting data with these time spans will form long-term, medium-term, and short-term traffic flow patterns with strong regularity and high similarity in the fluctuation curves. Based on the basic characteristics of urban road traffic flow, it is known that traffic flow data are time-series data, making time similarity analysis essential.

The modeling data used in this study were sourced from a single charging pile. The dataset includes the start and end times for each charging event, as well as the total energy consumed [13]. The raw dataset was converted into a corresponding dataset containing the hourly average charging load in kW—P(kWh). The data selected span from 1 January 2023 to 29 April 2023, with a prediction time period of one week, with an interval of one hour, totaling 2856 data points.

To thoroughly analyze the seasonal variation characteristics of the charging load, this study performed initial data preprocessing, including missing value imputation, outlier removal, and normalization, to ensure data completeness and consistency. Subsequently, the data were divided into the following three periods based on common seasonal classifications:

- Winter Period (1 January–3 February): During this period, low temperatures may lead to reduced battery performance and changes in user charging behavior, potentially affecting the charging load.
- Winter–Spring Transition Period (4 February–4 March): As temperatures gradually rise, the charging load may exhibit transitional characteristics.
- Spring Period (5 March–29 April): In this period, temperatures are moderate, and the charging load is likely to stabilize, reflecting typical spring user behavior patterns.

To assess the predictive performance of the model, the data for each period were split into training and testing sets in an 80:20 ratio. The training set was used for model training and hyperparameter optimization, while the testing set was employed to evaluate the final prediction accuracy and generalization capability of the model. This study, through seasonal partitioning and visualization analysis, comprehensively captured the seasonal variation patterns of the charging load. By training the model independently for each season, the influence of seasonal factors on the model was effectively reduced, allowing the

seasonal characteristics to be more accurately modeled, thereby enhancing the accuracy of the prediction results.

Figures 1–3 present the three-dimensional visualization results of the charging load data, segmented by season. In the figures, the x-axis represents 24 h of a day, the y-axis represents the days, and the z-axis represents the charging power (in kWh). The three-dimensional plots provide an intuitive observation of the spatial–temporal distribution characteristics of the charging load across different seasons.

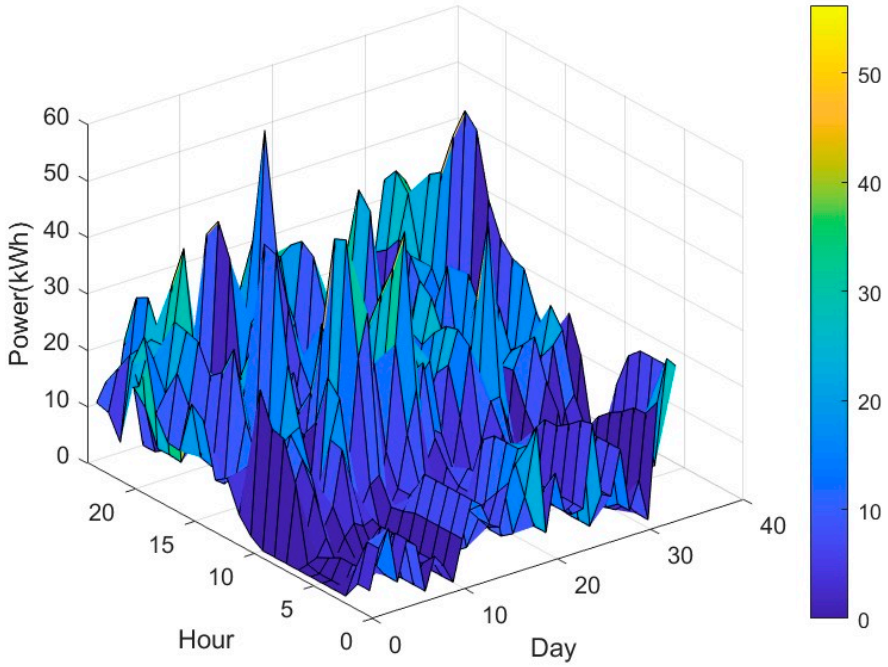

Figure 1. The three-dimensional spatial–temporal distribution of charging load in the winter season.

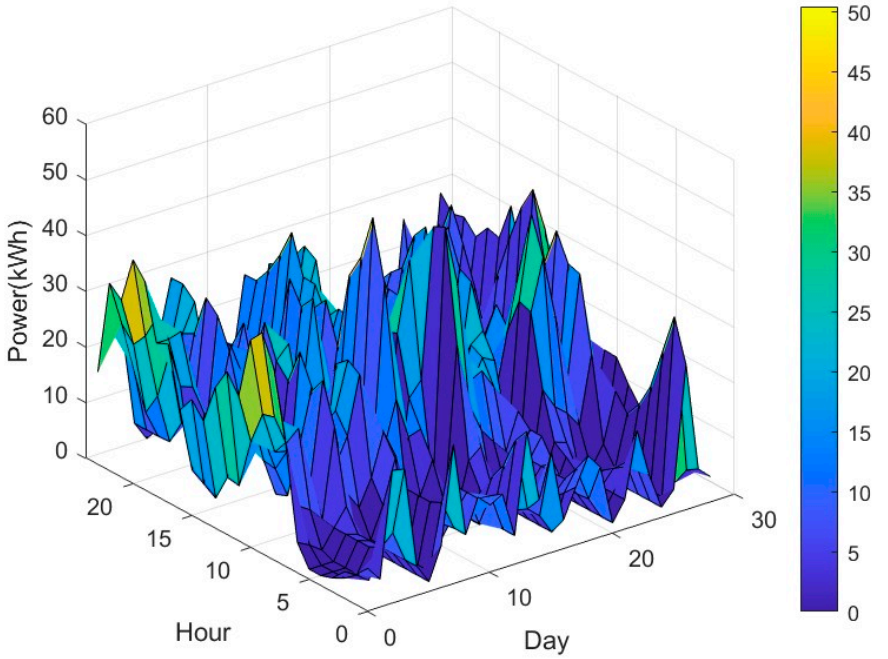

Figure 2. The three-dimensional spatial–temporal distribution of charging load during the winter–spring transition period.

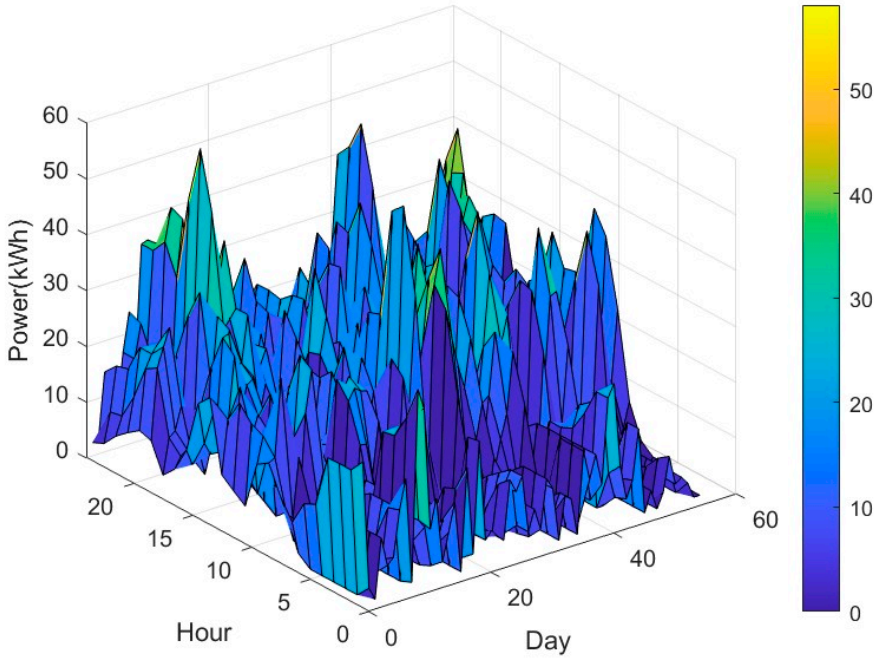

Figure 3. The three-dimensional spatial–temporal distribution of charging load in the spring season.

2.2. Improved LSTM Prediction Model

The Long Short-Term Memory (LSTM) network is a distinct variant of recurrent neural networks (RNNs) specifically designed to address the vanishing and exploding gradient problems typically encountered by conventional RNNs when processing long sequences of data. The core innovation of LSTM lies in its unique gating mechanism, which allows the network to dynamically regulate the flow of information. This mechanism enables the retention and transmission of critical temporal dependencies over long sequences [14].

The core structure of an LSTM unit is illustrated in Figure 4. It consists of three gating mechanisms—the forget gate, input gate, and output gate—along with a cell state update module. The forget gate (f_t), governed by a sigmoid function, determines which information to discard from the cell state (C_{t-1}). The input gate (i_t) and the candidate cell state ($\widetilde{C}_t$) collaboratively update the current cell state (C_t), while the output gate (o_t) regulates the output of the hidden state (h_t). The arrows indicate the direction of information flow, while the Tanh function serves as the activation function.

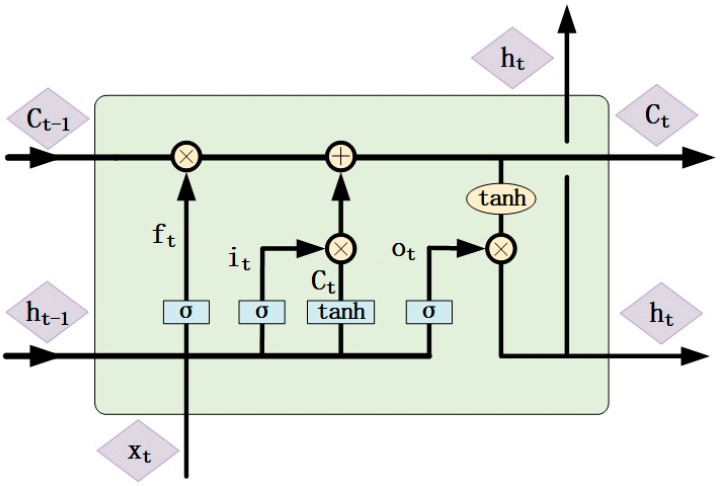

Figure 4. A structural diagram of the LSTM network.

2.2.1. Basic Structure of LSTM

The LSTM unit consists of three gates: the forget gate, the input gate, and the output gate. Each gate operates as a neural network layer, collaborating to determine which information should be retained, updated, or discarded.

1. Forget Gate:

It determines which information should be discarded from the cell state.

$$f_t = \sigma\Big(W_f \times [h_{t-1}, x_t] + b_f\Big) \tag{1}$$

where f_t represents the output of the forget gate, σ is the sigmoid function, W_f denotes the weight matrix, h_{t-1} is the hidden state from the previous time step, x_t is the current input, and b_f represents the bias term.

2. Input Gate:

It controls the storage of new information.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2}$$

$$\widetilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{3}$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \widetilde{C}_t \tag{4}$$

where i_t represents the output of the input gate, $\widetilde{C}_t$ is the new candidate value vector, and C_t is the current cell state.

3. Output Gate:

It determines the output of the hidden state.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{5}$$

$$h_t = o_t \cdot \tanh(C_t) \tag{6}$$

where o_t represents the output of the output gate, and h_t is the hidden state at the current time step.

The information propagation process in an LSTM neural network is described as follows [15]:

- Forgetting and Memory: The input information and stored information are multiplied by weight matrices, and after adding the bias term, they pass through a sigmoid function for normalization to obtain the final input information.
- New Information Input: During the input phase, the data are processed by passing them through the weight matrix and multiplying them with the activation matrix, producing the information that will be transferred to the memory unit.
- Cell State Update and Information Output: The results of the first two steps are combined to compute the current cell state. This cell state is then multiplied by the output matrix to generate the final output.

2.2.2. Improvements to the LSTM Model

To enhance the model's ability to capture patterns in time-series data, two key improvements have been made to the traditional LSTM model: the introduction of bidirectional LSTM and the stacking of multiple LSTM layers.

The standard LSTM model can only learn the features of the current time step from past time steps. In contrast, the bidirectional LSTM processes sequence data in both

forward and backward directions simultaneously, enabling it to capture comprehensive contextual information from both past and future time steps [16]. This design is particularly suitable for scenarios where there is a strong correlation between future and past time steps, such as in periodic load data. By integrating information from both time directions, bidirectional LSTM significantly improves the model's ability to understand the latent patterns in the sequence [17]. Stacking multiple LSTM layers allows the model to gradually extract higher-dimensional patterns, from shallow features to deeper ones.

This stacking approach mimics the structure of deep neural networks, progressively extracting more abstract features. During implementation, to ensure smooth information transmission across multiple layers, the first LSTM layer is set with "return_sequences = True" to output the complete time series for the next layer. The final LSTM layer is configured with "return_sequences = False" to produce a fixed-length vector as output, which is then connected to a dense layer for subsequent stages of prediction tasks.

The architecture of the improved LSTM model is shown in Figure 5. In this design, the bidirectional LSTM processes both forward and backward sequence data simultaneously, enabling the model to capture global contextual information between past and future time steps. The stacking of LSTM layers is as follows: The first layer consists of a bidirectional LSTM layer, with a dropout layer inserted in between to randomly deactivate neurons and reduce overfitting. The final LSTM layer outputs a fixed-length vector, which is then connected to a fully connected layer for the final load prediction. The input layer receives sequential data, with "x_0, x_1, x_2, . . ., x_i" representing the sequence elements. "A" and "A'" represent LSTM units processing in opposite directions, with red arrows indicating forward time steps and blue arrows denoting backward time steps. The sequence "h_0, h_1, h_2, . . ., h_i" represents the output hidden states.

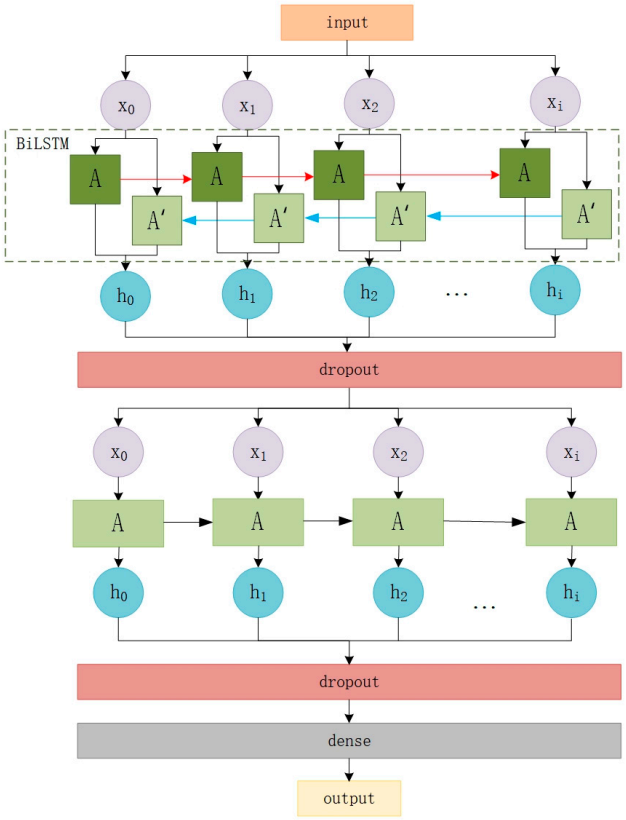

Figure 5. A diagram of the improved model structure.

2.3. LSTM Core Parameter Optimization Based on PSO Algorithm

The Particle Swarm Optimization (PSO) algorithm, introduced by Dr. Eberhart and Dr. Kennedy in 1995 [18], is a stochastic search method based on collective cooperation. It is inspired by the foraging behaviors of bird flocks. The algorithm's initial operational framework was designed as a simplified model of social behavior, particularly how birds forage in groups.

Researchers have found that individuals within a bird flock can interact and influence one another, a phenomenon that exists in biological groups as a mechanism for information sharing. The PSO algorithm leverages this information-sharing mechanism, allowing both individual particles and the entire swarm to synchronize their knowledge [19]. In the PSO algorithm, the term "particle" is used as an intermediary selection because the members of the group exist in actual states rather than abstract descriptions, such as mass, volume, speed, and acceleration. Therefore, particles are used as replacements. Particles communicate and exchange information, collectively forming a swarm.

The objective of the PSO algorithm is to guide multiple particles in finding the optimal solution within a multi-dimensional hypervolume, also known as the solution space, which serves as the search space for the optimization problem. In this framework, the optimal solution for each problem is represented as a particle within this space, and all particles are mapped into a D-dimensional space. Each particle has a fitness value, determined by an optimization function, which assesses the quality of its current position. Additionally, each particle is associated with a velocity vector that governs its movement and position. The particle swarm then searches for the current optimal particle within the solution space.

In the Particle Swarm Optimization (PSO) algorithm, each unit is referred to as a particle. A swarm consists of N particles, which are randomly initialized within a D-dimensional search space [20]. In the search process, each particle i is represented by two vectors: the velocity vector V_i and the position vector X_i. Each particle i updates its velocity and position using its personal best position $Pbest_i$ and the global best position $Gbest_i$ found so far. The update equations for X_i and V_i are as follows:

$$V_{id}^{t+1} = \omega * V_{id}^{t} + c_1 \cdot r_1 \cdot \left[Pbest_{id}^{t} - X_{id}^{t}\right] + c_2 \cdot r_2 \cdot \left[Gbest_{d}^{t} - X_{id}^{t}\right] \tag{7}$$

$$X_{id}^{t+1} = X_{id}^{t} + V_{id}^{t+1} \tag{8}$$

Here, X_{id}^{t} and V_{id}^{t} represent the velocity and position of particle i in dimension d during iteration t, respectively. $Pbest_{id}^{t}$ is the best position found by particle i in dimension d up to iteration t, while $Gbest_{d}^{t}$ is the best position found by the entire swarm in dimension d during iteration t. t and $t+1$ represent the current and next iterations, respectively [21]. ω denotes the inertia weight, and c_1 and c_2 are the acceleration coefficients for the cognitive and social components, respectively. r_1 and r_2 are random numbers uniformly distributed in the range [0, 1]. The personal best position and the global best position of each particle are updated in each iteration using the following equations:

$$Pbest_{id}^{t+1} = \begin{cases} X_{id}^{t+1} & , f\left(X_{id}^{t+1}\right) \le f\left(Pbest_{id}^{t}\right) \\ Pbest_{id}^{t} & , f\left(X_{id}^{t+1}\right) > f\left(Pbest_{id}^{t}\right) \end{cases} \tag{9}$$

$$Gbest_{d}^{t+1} = \begin{cases} Gbest_{d}^{t} & , f\left(Gbest_{d}^{t}\right) \le \min\limits_{i}\left(f\left(Pbest_{id}^{t+1}\right)\right) \\ Pbest_{id}^{t+1} & , f\left(Gbest_{d}^{t}\right) > \min\limits_{i}\left(f\left(Pbest_{id}^{t+1}\right)\right) \end{cases} \tag{10}$$

Figure 6 illustrates the concept of particle optimization, depicting the Particle Swarm Optimization (PSO) algorithm. The particle velocity update equation in the PSO algorithm

consists of three components. The first component represents the particle's inertia, reflecting its "memory" of the previous velocity, and is controlled by the inertia weight (ω). The second component captures the particle's self-awareness, driving it to move toward its own historical best position ($Pbest^t$), which reflects "self-awareness". Here, "X^{t+1}" denotes the adjusted position, and "V^{t+1}" represents the velocity increment. The third component represents the exchange of information and cooperation between particles, referred to as "socialization" [22], guiding particles toward the global best position ($Gbest^t$) and facilitating group information sharing.

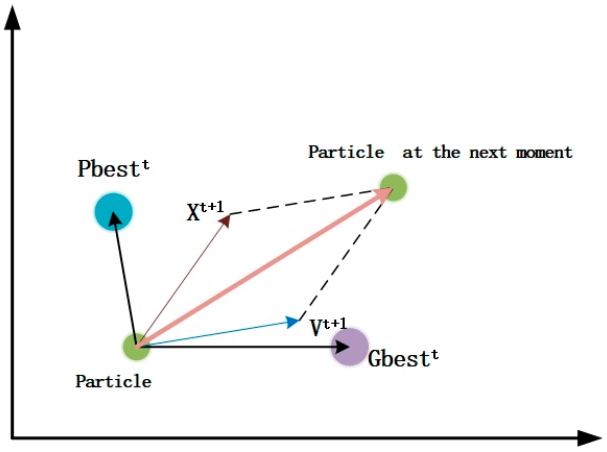

Figure 6. Diagram of particle global and historical optimal solutions, velocity, and position.

In applied research, the effectiveness of a model is primarily determined by the careful selection of key hyperparameters. However, traditional manual tuning methods are not only time-consuming and labor-intensive but also prone to getting stuck in local optima, preventing the model from fully realizing its potential. To address this issue, this study uses the Particle Swarm Optimization (PSO) algorithm to optimize five key parameters in the LSTM model: the number of neurons in the first hidden layer (Hidden Units1), the number of neurons in the second hidden layer (Hidden Units2), the dropout rate (Dropout), the batch size (Batch Size), and the number of training epochs (Epochs). Through the global search capabilities and efficient parameter adjustment strategy of the PSO algorithm, this study aims to identify the optimal combination of parameters to maximize the predictive performance of the LSTM model while minimizing resource consumption during the training process.

2.4. Development of the Electric Vehicle Load Forecasting Model

Identifying the optimal parameters for a predictive model is inherently challenging, and the training process is both time-consuming and computationally intensive. The optimal values obtained through manual tuning are often only locally optimal rather than globally optimal. To mitigate the errors and randomness introduced by manual tuning and to achieve automatic parameter optimization, this study employs a swarm intelligence optimization algorithm for parameter adjustment and optimization.

The predictive model in this study is based on the LSTM architecture, where key hyperparameters—such as the number of neurons, dropout rate, batch size, and number of training epochs—significantly influence both prediction accuracy and convergence speed. To enhance the model's performance, the Particle Swarm Optimization (PSO) algorithm is used to optimize these hyperparameters [23].

Figure 7 presents the overall framework of the PSO-LSTM model. Starting from the data preprocessing stage, the purple box section performs cleaning, missing value imputation, and normalization on the raw charging pile load data to ensure the quality

and consistency of the input data. Hyperparameter optimization plays a critical role in enhancing model performance. In the blue box section, the PSO algorithm is used to optimize the hyperparameters of the LSTM model, ensuring that the model identifies the globally optimal parameter combination within complex data patterns, thereby improving prediction accuracy and stability. The model construction phase is represented within the green box, where, based on the optimized parameters, a prediction model is built that includes a bidirectional LSTM layer, a dropout layer, and a fully connected layer. The bidirectional LSTM layer captures both forward and backward temporal dependencies in the data, while the dropout layer effectively prevents overfitting by randomly deactivating neurons. The fully connected layer integrates the extracted high-dimensional features and outputs the final prediction results. This framework not only significantly enhances the model's predictive performance but also provides reliable technical support for accurate load forecasting of electric vehicle charging stations.

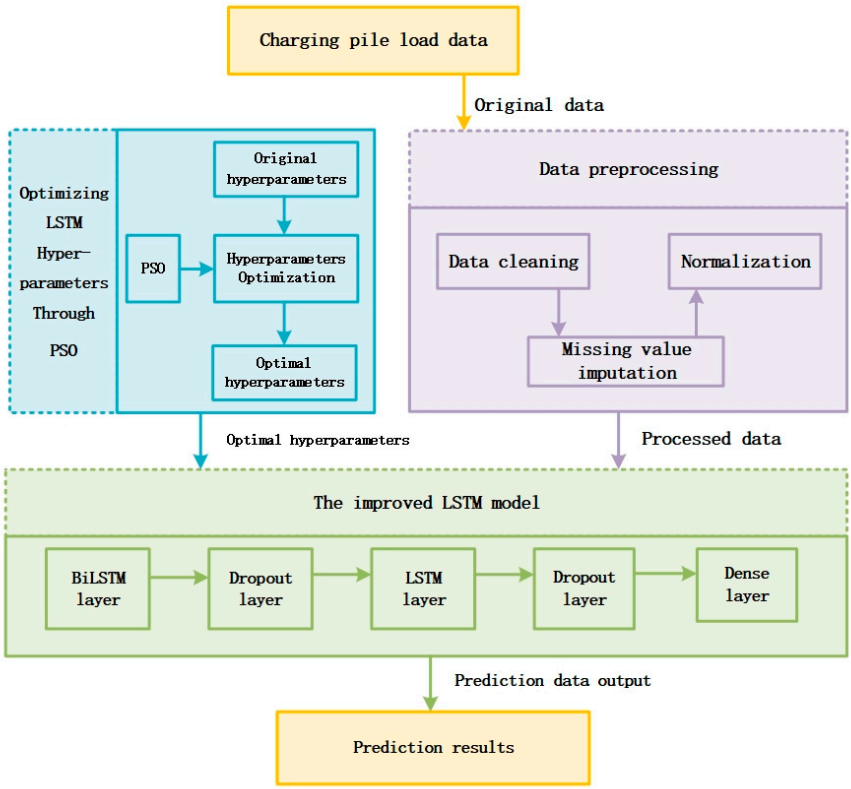

Figure 7. The overall framework of the PSO-LSTM model.

In the LSTM-based hybrid model, the network is structured with two layers: the first layer consists of two LSTM layers, while the second layer contains a single LSTM layer. Consequently, in the Particle Swarm Optimization (PSO) algorithm, the particle dimension is set to five, corresponding to five hyperparameters: the number of neurons in the first hidden layer (Hidden Units1), the number of neurons in the second hidden layer (Hidden Units2), the dropout rate (Dropout), the batch size (Batch Size), and the number of training epochs (Epochs).

Initially, the position range of the particles is defined by establishing the upper and lower bounds for the five dimensions. The Mean Squared Error (MSE) of the GRU-RNN model on the validation set is used as the fitness function. Subsequently, the fitness value of each particle is calculated, and the particles' velocity and position are updated according to the PSO update mechanism. The number of particles is set to 10, and the maximum number of iterations is set to 30, with both learning factors c_1 and c_2 set to 2.05.

The dataset used in this study, after undergoing missing value imputation, reorganization, and standardization, consists of a total of 2857 samples. Among them, 2352 samples are used in the validation set and fed into the model for parameter optimization. Once the optimization process is complete, the position of the global best particle in the five-dimensional space represents the optimal hyperparameters for the LSTM model.

The specific process for optimizing LSTM parameters using PSO is as follows:

Step 1: Initialize the parameters of the LSTM algorithm. Define the five hyperparameters to be optimized as the five-dimensional particle space. Initialize the position X and velocity V of each particle; set the number of particles, N; and specify the number of iterations, num.

Step 2: Define the fitness function as the mean squared error (MSE) of the LSTM model on the validation set. Compute the fitness values of all particles and identify the positions corresponding to the initial personal best and global best, with the lowest MSE values serving as the criteria.

Step 3: Update the velocity and position of each particle according to Equations (9) and (10), and then recalculate the fitness value for each particle.

Step 4: Update the individual best position and the global best position.

Step 5: Check whether the termination condition is met. If yes, output the global best position; if not, return to Step 3.

The final position of the global best particle in the five-dimensional space at the conclusion of the optimization process is considered the optimal set of hyperparameters for the LSTM model. Based on these optimized parameters, the PSO-LSTM predictive model is constructed. The initialization parameters for the PSO-LSTM model are shown in Table 3.

Table 3. Initialization of PSO-LSTM parameters.

Parameter	Initial Range
Population size, N	10
Number of iterations, num	30
Learning factor c_1	2.05
Learning factor c_2	2.05
Hidden Units1	64
Hidden Units2	32
Dropout	0.3
Batch size	32
Epochs	50

3. Experimental Results and Analysis

3.1. Experimental Setup

3.1.1. Experimental Environment Setup

The experiments were conducted on a Windows 10 operating system, utilizing the TensorFlow framework within the Anaconda environment. Programming was performed using the PyCharm editor. Anaconda is an open-source distribution that simplifies package management and environment management, offering built-in tools that do not require independent installation. It supports switching between different environments and is designed for the parallel use of multiple versions of Python [24]. TensorFlow is an open-source library commonly used for developing and implementing machine learning models, and it is regarded as one of the most popular frameworks in the field of machine learning. It includes numerous packages for scientific computing, image processing, and other tasks and supports automatic differentiation and model construction. PyCharm, developed by JetBrains, is a powerful and feature-rich Integrated Development Environment (IDE) specif-

ically designed for Python programming. It offers features such as code editing, debugging, testing, version control, and more, specifically designed for Python development, including powerful code autocompletion, code inspection, and one-click code navigation.

The specific experimental environment is shown in Table 4.

Table 4. Experimental environment.

Operating System	Windows 10
CPU	13th Gen Intel(R) Core (TM) i7-1360P 2.20 GHz
System RAM	32 GB
Learning framework	Tensorflow
Editor	PyCharm
Programming language	Python3.9

3.1.2. Model Parameter Settings

In this study, the PSO algorithm was used to fine-tune the key hyperparameters of the LSTM network, with the aim of enhancing its forecasting accuracy [25]. A two-layer LSTM network model was constructed, where the first layer consisted of a bidirectional LSTM to improve the model's ability to capture temporal dependencies in sequential data. The optimization process focused on five key hyperparameters: the number of neurons in the first hidden layer (Hidden Units1), the number of neurons in the second hidden layer (Hidden Units2), the dropout rate (Dropout), the batch size (Batch Size), and the total number of training epochs (Epochs).

To capture complex nonlinear relationships within input sequences, the search range for the number of neurons in the first hidden layer was set between 32 and 128, while the range for the second hidden layer was limited to 32 to 64 neurons. The dropout rate was optimized between 0.1 and 0.5 to balance learning capacity and generalization ability, aiming to prevent overfitting while maintaining model complexity. For the batch size, a range of 16 to 64 was selected to balance computational efficiency and model performance. Finally, the number of training epochs was set within the range of 10 to 100 to ensure adequate training and model convergence.

During the PSO algorithm's initialization, the population size, N, was set to 10, with each particle representing a possible hyperparameter combination. The fitness function was defined as the MSE of the LSTM model on the validation set, with the goal of minimizing MSE to assess the quality of various hyperparameter combinations. The number of iterations (num) was set to 30, indicating that the algorithm would run for a maximum of 30 iterations to identify the optimal solution. During the optimization process, the Mean Squared Error between predicted and actual values was used to guide the selection of the best hyperparameter combination at each iteration. To improve optimization performance, the inverse of the RMSE was incorporated into the fitness function, ensuring that the optimal neural network structure parameters identified during optimization were applied to construct the LSTM model.

To improve the accuracy of sequence prediction, this study standardized all subsequences and then input them into the PSO-LSTM model for training and forecasting. The training data were divided into three batches based on the seasons: winter, winter–spring transition, and spring. The data for each season were independently used to optimize the LSTM model, with the optimal hyperparameters determined through the PSO algorithm. After multiple iterations, the optimal hyperparameter combinations for each season were identified. Based on the optimization results from the PSO algorithm, the parameters for the PSO-LSTM model were configured, while the GRU and LSTM models used their default settings. Table 5 provides a detailed listing of the hyperparameter settings for all models.

Table 5. Parameter settings for all models.

Model	Hidden Units1	Hidden Units2	Dropout	Batch Size	Epochs
GRU	64	32	0.3	32	50
LSTM	64	32	0.3	32	50
PSO-LSTM (Winter)	115	57	0.2	16	43
PSO-LSTM (Winter–Spring Transition)	110	28	0.1	16	51
PSO-LSTM (Spring)	128	61	0.1	17	24

3.2. Comparative Analysis of Experimental Results

To comprehensively evaluate the model's prediction performance, this study employed three key metrics: Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). These metrics provide distinct perspectives on the difference between the predicted and actual values.

1. MAE is the average of the absolute errors:

   It determines which information should be discarded from the cell state.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{11}$$

where n is the number of samples, y_i represents the actual values, and $\hat{y}_i$ denotes the predicted values.

2. MSE is the average of the squared errors:

   It controls the storage of new information.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{12}$$

where n is the number of samples, y_i represents the actual values, and $\hat{y}_i$ denotes the predicted values.

3. RMSE is the square root of the Mean Squared Error (MSE):

   It determines the output of the hidden state.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{13}$$

where n is the number of samples, y_i represents the actual values, and $\hat{y}_i$ denotes the predicted values.

During the experiment, we independently trained and tested three different time-series forecasting models—GRU, LSTM, and PSO-LSTM—on the same dataset. This process was crucial for capturing the models' performance under different random initialization conditions, allowing for an accurate assessment of their stability [26]. Specifically, the training process for the GRU and LSTM models involved data normalization, dataset creation, network structure definition, and training loops. In contrast, the LSTM model optimized using PSO incorporated an additional phase, where the Particle Swarm Optimization algorithm was applied to fine-tune the parameters.

This study conducted three independent experiments using charging load data from different seasons, namely, winter, winter–spring transition, and spring, for model training and testing. Figures 8–10 present comparison curves between the real load values and the

predicted results from different models (GRU, LSTM, PSO-LSTM) for each season. In the figures, the black solid line represents the real load values, the green dashed line represents the predicted results from the GRU model, the blue dashed line represents the predicted results from the original LSTM model, and the red dotted line represents the predicted results from the PSO-optimized LSTM model.

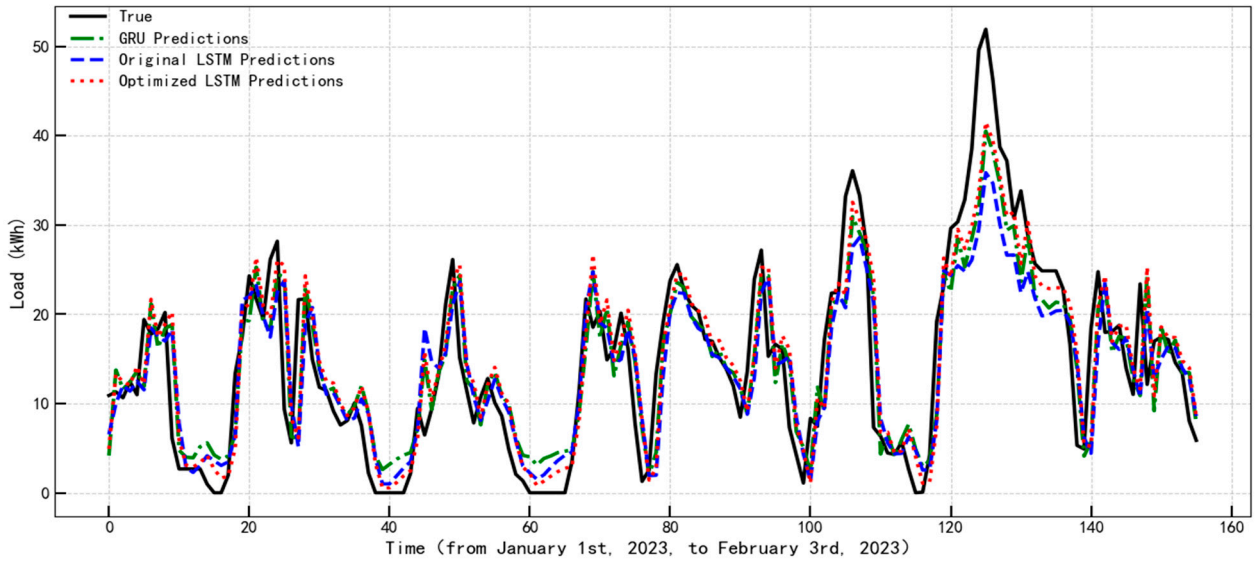

Figure 8. Winter charging load prediction comparison (1 January 2023–3 February 2023).

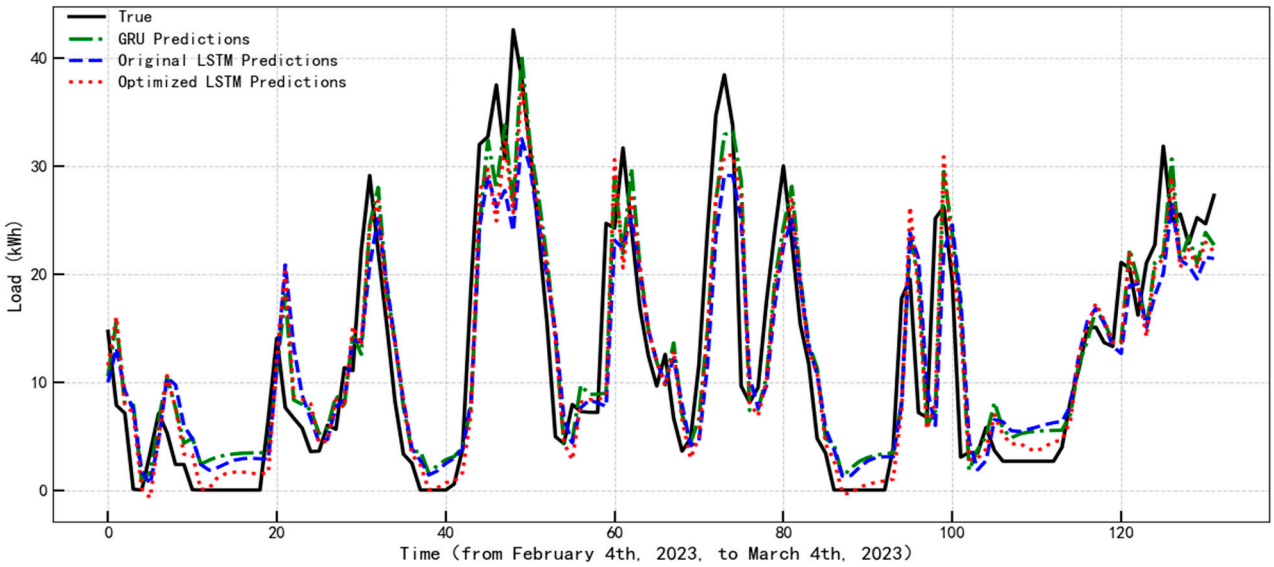

Figure 9. Winter–spring transition charging load prediction comparison (4 February 2023–4 March 2023).

The model output data were organized and summarized for evaluation metric analysis, with MAE, MSE, and RMSE calculated for each model. MAE provides a straightforward measure of the average absolute deviation between the model's predicted values and the true values. A smaller MAE indicates that the model's predictions have a smaller average error, meaning the predictions are closer to the actual values. MSE averages the squared prediction errors, amplifying the effect of larger errors and offering a more sensitive reflection of how well the model handles outliers. A smaller MSE suggests better overall prediction accuracy. RMSE, the square root of MSE, has the same units as the original data, making it easier to compare with the actual data and providing a more accurate measure of the average deviation between predicted and true values.

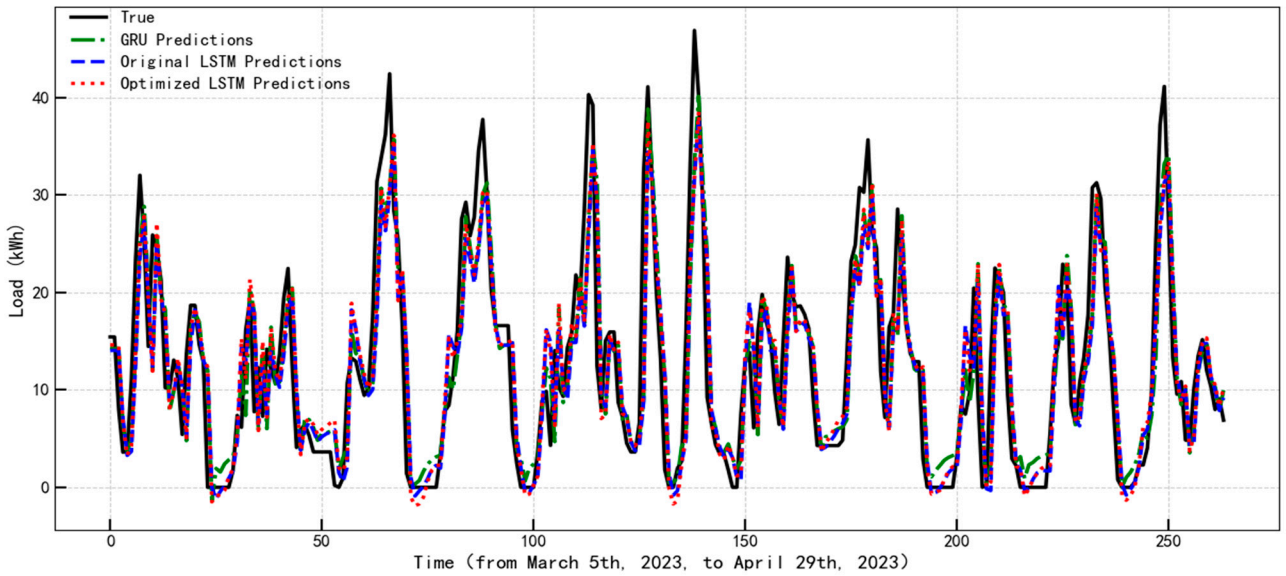

Figure 10. Spring charging load prediction comparison (5 March 2023–29 April 2023).

Table 6 summarizes all the evaluation metrics, which helps us gain insights from multiple dimensions into the strengths and weaknesses of each model under different seasonal conditions. This will provide a solid and reliable foundation for further model optimization and the development of charging load forecasting strategies.

Table 6. Comparison of model results.

Model	Season	MAE	MSE	RMSE
GRU	Winter	4.335	33.171	5.760
	Winter–spring transition	4.365	33.382	5.778
	Spring	4.219	31.046	5.572
LSTM	Winter	4.170	29.701	5.450
	Winter–spring transition	4.050	30.625	5.534
	Spring	4.019	31.292	5.594
PSO-LSTM	Winter	3.896	28.717	5.359
	Winter–spring transition	3.806	29.012	5.386
	Spring	3.910	29.796	5.458

Based on the performance metrics (MAE, MSE, RMSE) for different models across the seasons in Table 6, this study presents a detailed analysis from the following three perspectives:

- Winter Period (1 January–3 February): The MAE of the PSO-LSTM is 3.896, which is a reduction of 6.6% compared to LSTM (4.170) and 10.1% compared to GRU (4.335). Its RMSE (5.359) is significantly lower than those of both LSTM (5.450) and GRU (5.760), indicating stronger stability in high-variance load scenarios.
- Winter–Spring Transition Period (4 February–4 March): PSO-LSTM achieves the lowest MAE (3.806) and RMSE (5.386), demonstrating its adaptability to dynamic change patterns.
- Spring Period (5 March–29 April): Although the performance differences between models are narrow, the RMSE of PSO-LSTM (5.458) still outperforms GRU (5.572), indicating its competitiveness in stable scenarios.

From a comprehensive analysis of the three seasons' data, the GRU model consistently exhibits higher and more fluctuating error metrics across all seasons, with poor adaptability to seasonal changes, making it less effective in accurately capturing the variations in charging load across different seasons. The original LSTM model performs better than GRU

in terms of prediction accuracy but still falls short compared to PSO-LSTM in most cases, suggesting that although the LSTM model has some capacity for handling sequential data, it has limitations when facing complex seasonal charging load patterns. The PSO-LSTM model, however, maintains a low error level across all three seasons (winter, winter–spring transition, and spring), reflecting excellent generalization ability and adaptability to different seasonal data characteristics. Furthermore, it outperforms both GRU and original LSTM models in key evaluation metrics—MAE, MSE, and RMSE—demonstrating that the LSTM model optimized by the PSO algorithm offers superior performance and higher stability in load forecasting.

In conclusion, the LSTM model's powerful sequence modeling capability gives it an advantage in time-series forecasting. When combined with PSO algorithm-based parameter optimization, the model's performance is further enhanced, improving both prediction accuracy and computational efficiency. These findings not only validate the effectiveness of the LSTM model and PSO algorithm in time-series forecasting but also provide new directions for future research [27]. This involves examining the integration of various optimization algorithms with deep learning models and exploring their potential applications across a broader range of fields.

4. Conclusions

This study proposes a PSO-LSTM network model to enhance the accuracy and robustness of electric vehicle charging load forecasting. The LSTM model is trained using historical load data, and the PSO algorithm dynamically optimizes its key hyperparameters, such as the number of hidden layer neurons, dropout rate, batch size, and number of training epochs. Experimental results show that the PSO-LSTM model significantly outperforms traditional methods across different seasonal scenarios. For example, in winter, its MAE (3.896), MSE (28.717), and RMSE (5.359) are reduced by 10.13%, 13.43%, and 6.96% compared to the GRU model and by 6.57%, 3.31%, and 1.67% compared to the original LSTM model, validating the effectiveness of the PSO algorithm in parameter tuning. Additionally, the model maintains stable prediction performance in the winter–spring transition period and spring, demonstrating its ability to effectively capture seasonal load variation patterns and reduce the impact of climate and user behavior differences on forecast results, providing a highly adaptable solution for power system scheduling and charging facility planning.

Future research can expand in the following directions: First, by integrating meteorological data (e.g., temperature, humidity, wind speed) and traffic information, a multi-variable input model could be constructed to more comprehensively characterize the coupling relationship between charging behaviors and external environmental factors [28]. Second, a hybrid model architecture combining BiLSTM and a CNN could be developed, leveraging the bidirectional temporal modeling capability of BiLSTM and the spatial feature extraction advantages of CNN to further enhance the model's ability to analyze complex spatiotemporal patterns [29].

Author Contributions: Study conception and design: L.Z.; data collection: X.Y. and X.H.; analysis and interpretation of results: X.H. and L.Z.; draft manuscript preparation: L.Z. and X.Y.; draft manuscript editing and reviewing: X.H. and L.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by [the National Natural Science Foundation of China] grant number [61773243]; and [the Science and Technology Project of Chongqing Municipal Education Commission] grant number [KJZD-K202101702].

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors upon reasonable request.

Conflicts of Interest: The authors declare that they have no conflicts of interest.

References

1. Lee, Z.J. Large-Scale Adaptive Electric Vehicle Charging. In Proceedings of the 2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP), Anaheim, CA, USA, 26–28 November 2018; pp. 863–864. [CrossRef]
2. Wei, J.Z.; Ma, Z.P. Monte-Carlo-algorithm-based Load Prediction of Electric Vehicles Large-scale Charging. *Electr. Eng.* **2024**, *3*, 49–53. [CrossRef]
3. Shi, W.Q.; Wu, K.Y.; Wang, D.X. Eclectic Power System Short-Term Load Forecasting Model Based on Time Series Analysis and Kalman Filter Algorithm. *Control Theory Appl.* **2018**, *37*, 9–12+23.
4. Zhang, X.; Li, L. Seasonal Electric Vehicle Charging Load Prediction Based on Random Forest. *Softw. Eng.* **2024**, *27*, 11–14+37.
5. Song, M.S.; Li, Z.W.; Song, S. Research on the Optimization Strategy of Electric Vehicle OrderlyCharge and Discharge in Intelligent Community. *Tech. Autom. Appl.* **2022**, *41*, 17–22+27. [CrossRef]
6. Geng, P.; Yang, H.J.; Shi, Z.X. Electric Vehicle Forecasting Charging Demand Based on Spatiotemporal Graph Convolutional Networks. *J. Transp. Eng.* **2024**, *24*, 37–45.
7. Pei, Z.; Zhang, Z.; Chen, J. KAN-CNN: A Novel Framework for Electric Vehicle Load Forecasting with Enhanced Engineering Applicability and Simplified Neural Network Tuning. *Electronics* **2025**, *14*, 414. [CrossRef]
8. Liu, Y.X.; Gao, H. Load Prediction Method of Charging Station Based on SSA-VMD-BiLSTM Model. *Guangdong Electr. Power* **2024**, *37*, 53–61.
9. Lin, X.; Zhang, H.; Ma, Y.L. Electric vehicle charging load prediction based on improved LSTM neural network. *Mod. Electron. Tech.* **2024**, *47*, 97–101.
10. Huang, Y.X.; Xiao, S.W. Forecasting of electric vehicle charging load in highway service areas considering meteorological factors. *Appl. Energy* **2025**, *383*, 125337.
11. Yin, W.; Ji, J. Research on EV charging load forecasting and orderly charging scheduling based on model fusion. *Energy* **2024**, *290*, 130126. [CrossRef]
12. Ma, S.; Ning, J.; Mao, N.; Liu, J.; Shi, R. Research on Machine Learning-Based Method for Predicting Industrial Park Electric Vehicle Charging Load. *Sustainability* **2024**, *16*, 7258. [CrossRef]
13. Ge, Q.; Guo, C.; Jiang, H. Industrial power load forecasting method based on reinforcement learning and PSO-LSSVM. *IEEE Trans. Cybern.* **2020**, *52*, 1112–1124. [CrossRef] [PubMed]
14. Jin, Y.; Guo, H.; Wang, J. A hybrid system based on LSTM for short-term power load forecasting. *Energies* **2020**, *13*, 6241. [CrossRef]
15. Saoud, A.; Recioui, A. Load Energy Forecasting based on a Hybrid PSO LSTM-AE Model. *Alger. J. Environ. Sci. Technol.* **2023**, *9*, 2938–2946.
16. Liu, X.; Ma, Z.; Guo, H. Short-term power load forecasting based on DE-IHHO optimized BiLSTM. *IEEE Access* **2024**, *12*, 145341–145349. [CrossRef]
17. Lai, Y.; Wang, Q.; Chen, G. Short-term Power Load Prediction Method based on VMD and EDE-BiLSTM. *IEEE Access* **2024**, *13*, 10481–10488. [CrossRef]
18. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95-International Conference on Neural Networks, Perth, WA, Australia, 27 November—1 December 1995; Volume 4, pp. 1942–1948.
19. Liu, Z.; Chen, X.; Liang, X.; Huang, S.; Zhao, Y. Research on Sustainable Form Design of NEV Vehicle Based on Particle Swarm Algorithm Optimized Support Vector Regression. *Sustainability* **2024**, *16*, 7812. [CrossRef]
20. Dai, X.; Sheng, K.; Shu, F. Ship power load forecasting based on PSO-SVM. *Math. Biosci. Eng.* **2022**, *19*, 4547–4567. [CrossRef]
21. Geng, G.; He, Y.; Zhang, J. Short-term power load forecasting based on PSO-optimized VMD-TCN-attention mechanism. *Energies* **2023**, *16*, 4616. [CrossRef]
22. Fan, W.; Hu, Z.; Veerasamy, V. PSO-based model predictive control for load frequency regulation with wind turbines. *Energies* **2022**, *15*, 8219. [CrossRef]
23. Jain, M.; Saihjpal, V.; Singh, N. An overview of variants and advancements of PSO algorithm. *Appl. Sci.* **2022**, *12*, 8392. [CrossRef]
24. Kim, H.J.; Kim, M.K. Spatial-Temporal Graph Convolutional-Based Recurrent Network for Electric Vehicle Charging Stations Demand Forecasting in Energy Market. *IEEE Trans. Smart Grid* **2024**, *15*, 3979–3993. [CrossRef]
25. Güven, A.F. Integrating electric vehicles into hybrid microgrids: A stochastic approach to future-ready renewable energy solutions and management. *Energy* **2024**, *303*, 131968. [CrossRef]
26. Ding, L.; Ke, S.; Zhang, F. Forecasting of electric-vehicle charging load considering travel demand and guidance strategy. *Electr. Power Constr.* **2024**, *45*, 10–26.

27. Zhang, Q.; Lu, J.; Kuang, W.; Wu, L.; Wang, Z. Short-Term Charging Load Prediction of Electric Vehicles with Dynamic Traffic Information Based on a Support Vector Machine. *World Electr. Veh. J.* **2024**, *15*, 189. [CrossRef]
28. Qian, Y.; Kong, Y.; Huang, C. Review of Power Load Forecasting. *Sichuan Electr. Power Technol.* **2023**, *46*, 37–43+58. [CrossRef]
29. Zhang, X.W.; Liang, J.; Wang, Y.G.; Han, J. Overview of Research on Spatiotemporal Distribution Prediction of Electric Vehicle Charging. *Electr. Power Constr.* **2023**, *44*, 161–173.