


## Article

# Meta-Extreme Learning Machine for Short-Term Traffic Flow Forecasting

Xin Li <sup>1</sup>, Linfeng Li <sup>1</sup>, Boyu Huang <sup>1</sup>, Haowen Dou <sup>1</sup>, Xi Yang <sup>2,\*</sup> and Teng Zhou <sup>1,3</sup><sup>1</sup> Department of Computer Science, College of Engineering, Shantou University, Shantou 515063, China<sup>2</sup> School of Information Science and Technology, Guangdong University of Foreign Study, Guangzhou 510006, China<sup>3</sup> Key Laboratory of Intelligent Manufacturing Technology, Shantou University, Ministry of Education, Shantou 515063, China

\* Correspondence: xi\_yang@gdufs.edu.cn

**Abstract:** The traffic flow forecasting proposed for a series of problems, such as urban road congestion and unreasonable road planning, aims to build a new smart city, improve urban infrastructure, and alleviate road congestion. The problems encountered in traffic flow forecasting are also relatively difficult; the reason is that traffic flow forecasting is uncertain, dynamic, and nonlinear. It is challenging to build a reliable and safe model. Aiming at this complex and nonlinear traffic flow forecasting problem, this paper proposes a solution of an ABC-ELM model optimized by an artificial bee colony algorithm to solve the above problem. It uses the characteristics of the artificial bee colony algorithm to optimize the model so that the model can better and faster find the optimal solution in space. Moreover, it also uses the characteristics of the limit learning machine to quickly deal with this nonlinear specific problem. Experimental results on the Amsterdam road traffic flow dataset show that the traffic flow prediction model proposed in this paper has higher prediction accuracy and is more sensitive to data changes.

**Keywords:** artificial bee colony algorithm; extreme learning machine; short-term traffic flow forecasting

**Citation:** Li, X.; Li, L.; Huang, B.; Dou, H.; Yang, X.; Zhou, T. Meta-Extreme Learning Machine for Short-Term Traffic Flow Forecasting. *Appl. Sci.* **2022**, *12*, 12670. <https://doi.org/10.3390/app122412670>

Academic Editor: Luís Picado Santos

Received: 7 November 2022

Accepted: 5 December 2022

Published: 10 December 2022

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

The study of intelligent traffic management and control systems has become a hot topic in the construction of modern transportation systems. Among them, the key to achieving forward-looking traffic management and control is to achieve higher forecast accuracy for short-term traffic flows [1]. Short-term traffic flow forecasting is particularly important. Due to the time sequence of traffic flow, the practical application value of short-term forecasting is higher than that of long-term forecasting. A proven and effective short-term traffic flow forecasting model can improve traffic quality in several ways, including reducing travel costs [2], reducing carbon emissions due to traffic congestion, and improving the efficiency of traffic operations [3]. Currently, short-term traffic flow forecasting techniques are applied in many domains, including transit routing models [4], rail transit ridership [5], and dockless shared bike demand [6]. However, it is challenging to build an accurate and stable short-term traffic flow forecasting model because of the nonlinear and stochastic characteristics of the traffic flow [7]. In addition, other external factors can also interfere with the accuracy of the forecast, such as unanticipated traffic accidents or unplanned traffic controls.

During the past decades, researchers have contributed a series of new theories and methods in the field of traffic flow forecasting. Traffic flow forecasting is broadly grouped into three categories, namely, traditional machine learning models, dynamics-based models, and deep learning models. First, traditional machine learning models include *k*-nearest neighbor (*k*NN) [2], support vector machine (SVR) [8], ensemble learning [9], etc. For example, Cui et al. proposed a two-stage hybrid model to determine the combination

of hyperparameters for the extreme learning machine [10]. Wei et al. proposed a noise-immune extreme learning machine, which takes advantage of the gravitational search algorithm (GSA) to search for an optimal global solution and used the extreme learning machine (ELM) to forecast traffic flow [11]. Cai et al. proposed a hybrid model using the GSA to optimize SVR for short-term traffic flow forecasting in [12]. Cai et al. also proposed using KNN regression models for short-term traffic flow forecasting [2]. Moreover, an ELM model with the particle swarm optimization (PSO) algorithm was proposed by Cai et al. [13]. An extreme gradient boosting-based learning model enhanced by wavelet denoising was proposed by Zheng et al. [14], which was easy to implement and effective.

An ELM optimized by GSA for short-term traffic flow prediction [15] was proposed by Cui et al. Second, dynamics-based models include auto-regressive integrated moving average (ARIMA) [16–18], historical average (HA), Kalman filter (KF) models [3,19,20], exponential smoothing (ES), seasonal auto-regressive integrated moving average [21,22], etc. For example, Zhang et al. designed a KF for accurate traffic flow forecasting by identifying the noises in the traffic flow data [19]. Zhou et al. proposed a hybrid dual KF for short-term traffic flow forecasting [3]. Furthermore, Cai et al. proposed embedding a fixed-point algorithm to update the posterior estimate of the KF, which is derived from maximum correlation entropy,

To solve the problem that the KF derived from the maximum correlation entropy criterion is insensitive to non-Gaussian noise [20]. From the 20th century to the present, artificial intelligence and machine learning have made significant progress. Deep learning models have been applied in the field of complex nonlinear short-term traffic flow forecasting, including deep belief network [23], fuzzy logic system methods [24,25], deep feature fusion model [26], etc. Zhou et al. proposed training stacked autoencoders to improve the accuracy of traffic flow forecasting [27]. A novel multi-diffusion convolution block constructed from the diffusion convolution of attention mechanism and bidirectional diffusion convolution was proposed by Lu et al [28]. Lu et al. also proposed a novel long short-term memory (LSTM) network enhanced by temporal-aware convolutional context (TCC) blocks and a new loss-switch mechanism (LSM) in [29]. In addition, Fang et al. proposed making an extension to the LSTM neural network by introducing the attention mechanism into it for short-term traffic flow prediction [30,31]. The advantages of neural networks include flexibility, strong ability to deal with nonlinear and complex problems, and parallel computing [32–34], but there are still shortcomings. For example, as a non-parametric method of the ANN, its parameters change with data points. The forecasting accuracy of the ANN is limited by the sample size. Therefore, to train a successful ANN model, a huge amount of raw data is required [35,36]. Various prediction schemes have been mentioned above, but they have shortcomings. We would like to propose a new model to improve on the shortcomings as well as the accuracy of traffic flow prediction.

In summary, the task of traffic flow forecasting is to train a network model  $g$  whose role is to accurately map traffic flow data  $T$  to future traffic flow data  $P$ , e.g.,  $P = g(T)$ . However, there is some difficulty in finding and training the optimal mapping function  $g^*$  that is adapted to all datasets and achieves the best performance. In order to address these issues, we rethink the potential of evolutionary algorithms to develop models that embed a model capable of automatic optimization in a traffic flow prediction model, resulting in a new hybrid model. For example, a model  $g$  that automatically determines the best parameters in a basic forecasting model  $G$  of traffic flow data is expressed as  $g^* = G(X, g)$ . In this case, we take into account the full potential of optimization algorithms and obtain an effective model that can be optimized in a data-driven fashion over and over again by a simple hybrid of optimization algorithms and forecasting models.

In this paper, we develop a short-term traffic flow forecasting model called ABC-ELM, employing the artificial bee colony (ABC) algorithm model embedded into the ELM model. The ELM was first proposed in 2004 by Huang et al. [37]. In the ELM model, during initialization the generation of input weights and hidden layer deviations are random; then, the excitation function is calculated through the input data. The Moore–Penrose (MP)

generalized calculation inverse calculates the output matrix weights [38]. This characteristic not only makes it much faster than traditional learning models, but also avoids problems in traditional learning models, such as falling into a local optimum, learning rate too large or too small, etc. [39]. However, it also has drawbacks as it is randomly determined that the prediction accuracy of the destined ELM will not be particularly good, and there is an overfitting problem. To solve this problem, we propose embedding and incorporating the ABC algorithm model in the ELM. The ABC algorithm selects the optimal parameters in the process of random parameter determination in the ELM. Currently, ABC is mostly used to optimize training neural networks, such as the hybrid algorithm Ozturk et al. proposed in [40], which combines the ABC algorithm and the Levenberg-Marquardt (LM) algorithm, after which the hybrid algorithm is used to train an ANN. In this way, based on the experimental results of predecessors, we do not need to increase or delete the parameters of ABC-ELM to improve the accuracy of the model and at the same time prevent the overfitting problem caused by too many parameters. The main contributions of the ABC-ELM model proposed in this paper for short-term traffic flow forecasting include the following points:

- First, we reconceptualize the improvement of traffic flow forecasting models from a *meta* model perspective and exemplify a data-driven optimization algorithm to optimize the learning model.
- Second, based on the ELM end-to-end mechanism principle, we propose a combination of the ABC algorithm and the ELM.
- Third, through sufficient experiments on four benchmark datasets, the ABC-ELM model outperforms in comparison to the most advanced models.
- Fourth, the ABC algorithm used in the ELM not only did not increase the model complexity of the ELM but also unleashed the potential of ELM in short-term traffic flow forecasting.

The rest of the paper is divided into different parts. The second part is the methodology, and the third part is the experiment, which shows our results about the effect of the ABC-ELM algorithm. Finally, the conclusion follows.

## 2. Methodology

Since the extreme learning machine was proposed, it has achieved good results in regression and multiclass classification fields [41]. So, in this section, we introduce the relevant content of the ABC-ELM model, including the parts of ELM and ABC, followed by the ABC algorithm for performance optimization of the ELM model.

### 2.1. Artificial Bee Colony Algorithm

As the name suggests, the mechanism of the artificial bee colony algorithm is designed and created by imitating the behavior of a bee colony collecting honey. It is one of the algorithms based on swarm intelligence [42], and it was first proposed by Karaboga et al. in 2005. As a cluster intelligent optimization algorithm, its characteristic is that it has a faster convergence speed and does not easily fall into the local optimum. In the iterative optimization process, the algorithm does not need to know the special information of the problem; it only needs to compare each solution which generates the advantages and disadvantages. The global optimization is finally achieved through the local optimization behavior of the individual artificial bee.

Assuming that the problem has  $D$  characteristic parameters, i.e., the solution space is  $D$ -dimensional, the ABC algorithm regards the solution process of the optimization problem as an optimization problem in the dimensional search space. Each nectar is a position in the  $D$ -dimensional space, representing a possible solution to the problem. The nectar volume of the nectar source corresponds to the suitability of the corresponding solution [43].

During initialization, we assume that the number of both collecting and observing bees is  $N$ , the number of collecting bees and the number of observing bees are equal to the number of nectar sources, and one collecting bee corresponds to one nectar source. The

honey bee corresponding to the  $i$ th nectar source searches for a new nectar source according to the following formula:

$$x'_{id} = x_{id} + \phi_{id}(x_{id} - x_{kd}), \quad (1)$$

in which  $\phi_{id}$  is the acceleration factor that is randomly taken from the interval  $-1$  to  $1$ , but  $k \neq i, i = 1, 2, \dots, N, d = 1, 2, \dots, D$ . The new solution  $X'_i = \{x'_{i1}, x'_{i2}, \dots, x'_{iD}\}$  generated by each iteration calculates its fitness and then compares it with the adaptation value of the currently saved optimal solution  $X_i = \{x_{i1}, x_{i2}, \dots, x_{iD}\}$  and retains better solutions through the greedy selection strategy. After collecting bees to find the source of nectar, observing bees look for new possible solutions near the corresponding nectar source of the collecting bees. Each observing bee selects a nectar source based on probability, and the probability formula is

$$p_i = \frac{fit_i}{\sum_{i=1}^N fit_i}, \quad (2)$$

where  $fit_i$  is the fitness of possible solution  $X_i$  if the fitness value of a nectar source does not increase within the given upper bound of abandonment  $L$ . After all collecting bees and observing bees have searched the entire space, the nectar source is abandoned, and the nectar bee corresponding to the nectar source becomes a scout bee. They search for new possible solutions by the following formula

$$x_{id} = x_d^{min} + \tau(x_d^{max} - x_d^{min}), \quad (3)$$

where  $\tau$  is a random number in the interval  $0$  to  $1$ ,  $x_d^{max}$  and  $x_d^{min}$  are the decision variables upper bound and lower bound of  $d$ th dimension. A more detailed algorithm flow of the ABC algorithm is shown in Algorithm 1.

---

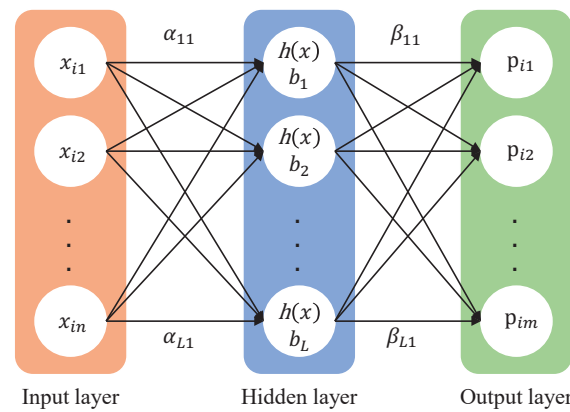
#### Algorithm 1 Artificial Bee Colony(ABC) algorithm

---

- 1: Set ABC parameters;
  - 2: Initialize food source location;
  - 3: Construct the current global optimal solution BestSol;
  - 4: **while** Reach the maximum number of iterations *MaxIt* **do**
  - 5:   **for** All employed bees **do**
  - 6:     Select food source according to Roulette Wheel Selection algorithm;
  - 7:     Search for a new food source near the corresponding food source  $pop(i)$ ;
  - 8:     Calculate fitness values for new food sources  $newbee.Cost$ ;
  - 9:     **if** The fitness value of the original food source  $pop(i).Cost \geq newbee.Cost$  **then**
  - 10:        $pop(i) = newbee$ ;
  - 11:     **end if**
  - 12:   **end for**
  - 13:   **for** All following bees **do**
  - 14:     Search for a new food source near the corresponding food source  $pop(i)$ ;
  - 15:     calculate fitness values for new food sources  $newbee.Cost$ ;
  - 16:     **if** The fitness value of the original food source  $pop(i).Cost \geq newbee.Cost$  **then**
  - 17:        $pop(i) = newbee$ ;
  - 18:     **end if**
  - 19:   **end for**
  - 20:   Compare and update the global optimal solution BestSol;
  - 21:   **for** All detecting bees **do**
  - 22:     **if** neighborhood search times  $C(i) \geq L$  **then**
  - 23:       reinitialize food source  $pop(i)$ ;
  - 24:        $C(i) = 0$ ;
  - 25:     **end if**
  - 26:   **end for**
  - 27: **end while**
-

### 2.2. Extreme Learning Machine

The extreme learning machine was first proposed in 2004 by Huang et al. [37]. During the training of traditional feedforward neural networks, which uses the gradient descent algorithm, its parameters tend to be the optimal solution in the update and calculation of each iteration [44]. However, the traditional feedforward neural network has many shortcomings, such as slow learning speed and complex network architecture. Therefore, the original intention of the extreme learning machine is to overcome these problems. ELM is regarded as a special kind of FNN in research or an improvement of FNN and its backward propagation algorithm [45]. The ELM architecture is divided into three layers, the input layer, the hidden layer, and the output layer, as shown in Figure 1. Different from the traditional feedforward neural network, the hidden layer of the extreme learning machine has only one layer. It is a single-layer feedforward neural network algorithm; its characteristic is that the input weights and biases of the hidden layer nodes are randomly generated or artificially given [13]. The learning process only needs to calculate the output weight through two; this feature has long meant that it has a faster learning speed and higher generalizability, as well as a strong learning and fitting ability.



**Figure 1.** The three layers of an extreme learning machine with one hidden layer in which  $\alpha$  denotes the input weights,  $\beta$  denotes the output weights, and  $b$  is the hidden layer deviations.

Suppose there are  $N$  arbitrary samples  $(X_i, t_i)$ , where  $i = 1, \dots, N$ ,  $X_i = [x_{i1}, x_{i2}, \dots, x_{in}]^T \in \mathbb{R}^n$ ,  $t_i = [t_{i1}, t_{i2}, \dots, t_{im}]^T \in \mathbb{R}^m$ . For a single hidden layer neural network with a hidden layer node, it can be expressed as

$$\sum_{i=1}^L \beta_i g(W_i \cdot X_j + b_i) = p_j, j = 1, \dots, N, \tag{4}$$

where  $\beta_i$  is the connection output weights between the hidden layer and output layer,  $g(x)$  is the activation function,  $W_i = [\alpha_{i1}, \alpha_{i2}, \dots, \alpha_{in}]^T$  is the connection weights between the input layer and the hidden layer,  $b_i$  is the bias of the  $i$ th hidden neuron,  $W_i \cdot X_i$  express the inner product of  $W_i$  and  $X_i$ , and  $p_j$  is the forecasting output value. The purpose of the extreme learning machine is to minimize the output error. It can be expressed as

$$\sum_{j=1}^N \|p_j - t_j\| = 0, \tag{5}$$

where  $\|$  is the Frobenius norm of the matrix elements, and Equation (5) means  $W_i$ ,  $b_i$ , and  $\beta_i$  exist and makes

$$\sum_{i=1}^L \beta_i g(W_i \cdot X_j + b_i) = t_j, j = 1, \dots, N, \tag{6}$$

its matrix, which is expressed as

$$H\beta = T, \tag{7}$$

where  $H$  is the output matrix of the hidden layer,  $\beta$  is the output weights, and  $T$  is the expected output, where  $\beta = [\beta_1^T, \beta_2^T, \dots, \beta_N^T]^T$ ,  $T = [t_1^T, t_2^T, \dots, t_N^T]^T$ ,  $H(W_1, \dots, W_L, b_1, \dots, b_L, X_1, \dots, X_L) = \begin{bmatrix} g(W_1 \cdot X_1 + b_1) & \cdots & g(W_L \cdot X_1 + b_L) \\ \vdots & \ddots & \vdots \\ g(W_1 \cdot X_N + b_1) & \cdots & g(W_L \cdot X_N + b_L) \end{bmatrix}_{N \times L}$ .

In the ELM algorithm, once the input weight  $W_i$  and the bias  $b_i$  of the hidden layer are randomly determined, the output matrix  $H$  of the hidden layer can be uniquely determined. Training a single hidden layer neural network can be transformed into solving a linear system  $H\beta = T$ , and the output weight can be determined by the following equation

$$\hat{\beta} = H^\dagger T, \tag{8}$$

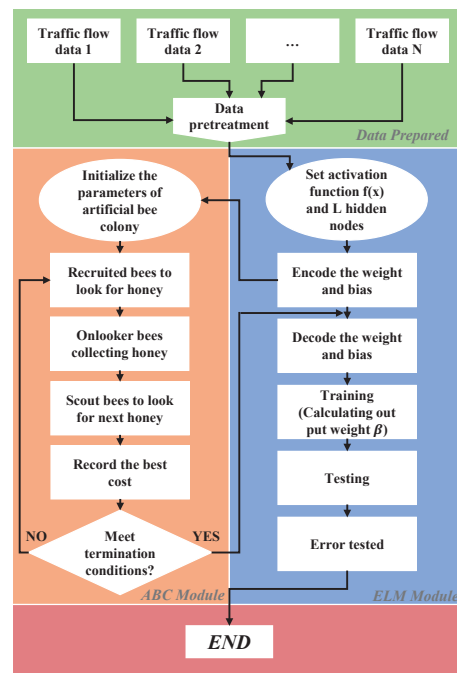
where  $H^\dagger$  is the Moore–Penrose generalized inverse of matrix  $H$ .

### 2.3. ABC-ELM for the Traffic Flow Forecasting

Because the input weight matrix and hidden layer bias of extreme learning machines are generated randomly in traditional extreme learning machines, if some values of input weights or hidden layer biases are 0 during the process of random assignment, the hidden layer nodes will be invalid. Thus, in some practical applications or large-scale training that needs a lot of nodes of hidden layers to ensure forecasting accuracy, it is difficult to ensure that so many hidden layer nodes achieve the expected accuracy. Moreover, when dealing with the samples that do not appear in the training process, a traditional extreme learning machine has insufficient generalization ability [46].

The traditional extreme learning machine model with randomly taken parameters brings limitations, yet there is room for optimization. To address this limitation, we propose an artificial bee colony algorithm to find the optimal network parameters for the ELM traffic flow prediction model. We use the artificial bee colony algorithm to continuously iterate to find the optimal solution. We construct and integrate the ABC framework to find the optimal input weight matrix and hidden layer bias for ELM in an iterative approach, which is called ABC-ELM in the following. Then, we find the optimal solution of the parameters of ELM by calculating the mean absolute percentage error (MAPE) and root mean square error (RMSE) in each iteration of the ABC algorithm. After that, we compare the performance of previous iterations and validate the performance of the ABC-ELM framework by cross-sectional comparison with other models. The workflow of the ABC-ELM is shown in Figure 2.

At the beginning of the algorithm, we performed initialization operations, including building the ELM network and initializing the ABC algorithm. First, for the ELM part, there are two main parameters to be initially selected, the activation function  $f(x)$  and the number of nodes in the hidden layer  $k$ . The higher the number of nodes matrix, the more accurate the prediction result is. An increase in the number of nodes also increases the training cost, and the activation function uses the sigmoid function. Secondly, for the artificial bee colony algorithm, we specify the solution space size, upper and lower boundaries of the decision variables, maximum number of iterations, colony size, etc. The solution space size is determined by the ELM input matrix  $W$  and the implicit layer parametric matrix  $B$ . Here, we set the size of the solution space to  $k \times N$ . Because the shape of  $W$  is  $[k, N]$  and the shape of  $B$  is  $[k, 1]$ , we compress both into a column vector with the shape  $[k \times N, 1]$ , where  $N$  is the number of training samples, and  $k$  is the number of nodes in the hidden layer of ELM. The pseudo-code of ABC-ELM is shown in Algorithm 2.



**Figure 2.** The workflow of ABC-ELM includes data pretreatment, ELM module, and ABC module. The ABC module passes the optimal solution back to the ELM module, where the predictions and calculations are performed.

#### Algorithm 2 ABC-ELM

- 1: Initialize the ELM network;
- 2: Set activation function  $f(x)$  and the number of hidden nodes  $k$ ;
- 3: Initialize the Artificial Bee Colony;
- 4: Determine colony size  $n$ , the maximum of iterations  $I$ , the upper bound of abandon  $L$  and the acceleration coefficient upper bound  $a$ ;
- 5: Data pre-processing;
- 6: Input the processed samples;
- 7: **while** termination conditions **do**
- 8:   **for** each  $i \in [1, I]$  **do**
- 9:     **for** each  $i \in [1, n]$  **do**
- 10:       Recruited Bees Stage;
- 11:       finding new positions randomly;
- 12:       evaluation;
- 13:       calculate fitness values and selection probabilities;
- 14:     **end for**
- 15:     **for** each  $i \in [1, n]$  **do**
- 16:       Onlooker Bees Stage;
- 17:       finding new positions with roulette algorithm;
- 18:       evaluation;
- 19:     **end for**
- 20:     **for** each  $i \in [1, n]$  **do**
- 21:       Scout Bees Stage;
- 22:       Discard more than  $L$  honey source and search for new;
- 23:     **end for**
- 24:     Update and store best solution ever found;
- 25:   **end for**
- 26: **end while**
- 27: Get input weight  $w$  and hidden layer bias  $b$ ;
- 28: Calculate output weight  $\beta$  of hidden layer;
- 29: Test and Error calculation;

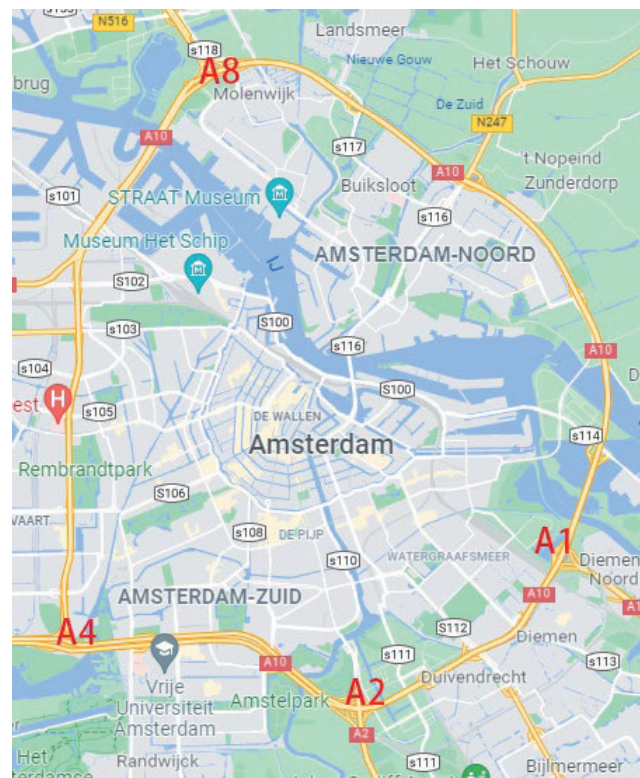
### 3. Experiments

In this section, the training data come from the four intersections of the A10 roundabout in Amsterdam, namely A1, A2, A4, and A8, which enter the roundabout. These data are collected at these four points of interest (i.e., the four entry points) via roadside units and uploaded to the traffic control center in real time [47].

#### 3.1. Description of Data

These data were collected by Wang et al. [48]. The data collection locations A1, A2, A4, and A8 are the converging intersections of the loop main road A10, as shown in Figure 3. The data were taken from 20 May to 24 June 2010 and were collected at four road junctions in 1 min aggregation via roadside units, which represent the number of vehicles at the junction at that moment.

In the original data, there are some wrong data that are zero or negative, which will cause anomalies in the model learning. So, we replace these wrong data by averaging the data from different dates in the same time period.



**Figure 3.** The four intersections of the Amsterdam A10 ring road that access the ring road, named A1, A2, A4, and A8. Sensors are embedded in the RSU and placed at each intersection.

#### 3.2. Experimental Setup

The focus of short-term traffic flow forecasting is not precisely on the ups and downs of the number of vehicles per minute [1,27,48], but on periods with certain time intervals. Therefore, we calculate the average value of vehicles per ten minutes based on the aggregated data of every minute in the original data, which is preprocessed and used as the sample data for the input model.

The data includes a total of 5 weeks of data, which is divided into two parts in this paper; the data of the first 4 weeks form the training set, and the data form the last week as the test set. Since the amount of sample data is not very large, including 5040 data in total, to ensure the reliability of the prediction accuracy the sliding window fetching method is used to process the input samples. Each input vector of the ABC-ELM prediction model consists of 9 consecutive data values of the sample data, and the 10th data value



is used as the corresponding output prediction. Moreover, we set 50 hidden layer neuron nodes for ELM and conducted a pre-experiment with 200 iterations, and the convergence performance is shown in Figure 4. From the figure, we can see that the model reaches the convergence state when the number of iterations reaches 100 or more. So, we think the maximum number of 200 iterations is a more reasonable setting. More detailed parameter value settings are shown in Tables 1 and 2.

Table 1. ABC parameters settings.

Abbreviation	Description	Value
nVar	number of decision variables	500
varSize	decision variables matrix size	(1500)
varMin	decision variables lower bound	-1
varMax	decision variables upper bound	1
maxIt	maximum number of iteration	200
nPop	population size(colony size)	100
nOnlooker	number of onlooker bees	100
a	acceleration coefficient upper bound	1.2

Table 2. ELM parameters settings.

Abbreviation	Description	Value
N	number of hidden layer nodes	50

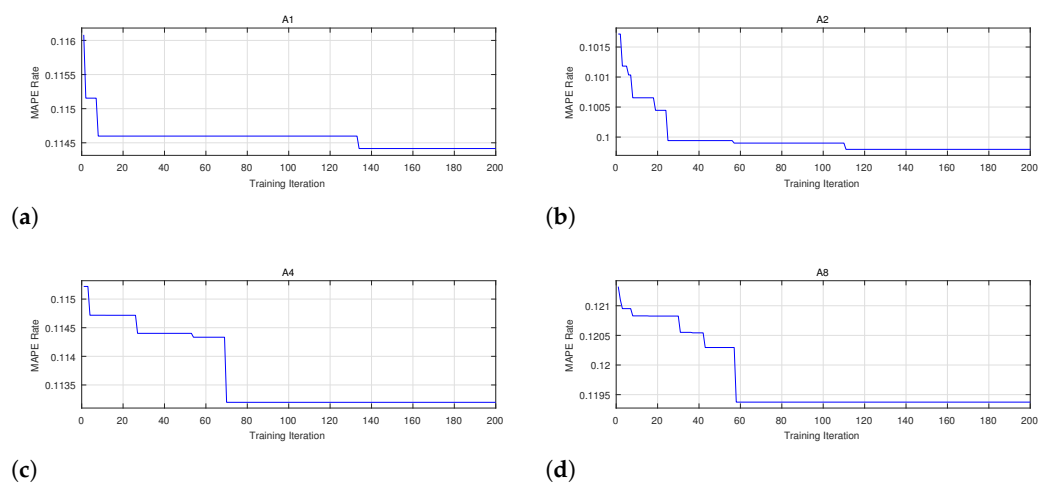


Figure 4. The (a–d) were predicted by the ABC algorithm on A1,A2,A4,A8 datasets with different numbers of iterations. The MAPE of the function tends to stabilize when the number of iterations exceeds 140.

### 3.3. Evaluation Criterion

Two evaluation criteria were used in this experiment to measure the performance of the ABC-ELM model. The first one is the root mean square error (*RMSE*), which is the square root of the ratio of the square of the deviation of the predicted value from the true value to the number of observations *N* and can reflect the precision of the measurement well. The second one is the mean absolute percentage error (*MAPE*), which has stronger robustness than *RMSE* because it is less susceptible to individual outliers. Although *RMSE* is very sensitive to the presence of outliers in the sample, this characteristic causes *RMSE* to be non-robust and susceptible to large errors. Therefore, this paper focuses on comparing

*MAPE*, and *RMSE* as used as an auxiliary evaluation index. These two evaluation criteria are expressed as Equations (9) and (10), respectively.

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N [\hat{v}(i) - v(i)]^2} \quad (9)$$

$$MAPE = \frac{1}{N} \sum_{i=1}^N \left| \frac{\hat{v}(i) - v(i)}{v(i)} \right| \times 100\% \quad (10)$$

where  $\hat{v}(i)$  and  $v(i)$  represent the forecasting and the true measurement of the  $i^{th}$  sample, and  $N$  is the total number of input samples.

### 3.4. Performance Evaluation

We evaluated the performance of our ABC-ELM by comparing the performance of models which are mostly applied to forecasting under the same dataset and reflected by the *RMSE* and *MAPE* evaluation criteria, as shown in Tables 3 and 4. Common models applied to linear regression forecasting include:

- Support Vector Machine (SVM) was proposed by Vapnik et al. [49]. The setting of SVR can be found at [27]. SVM performs nonlinear classification by a kernel method, which is one of the common kernel learning methods [50].
- Autoregression (AR) is a linear regression model widely cited for forecasting. Especially in the field of applied traffic flow, AR can perform well in the field of traffic flow, which is full of randomness, because it expresses the random variables of the latter through a linear combination of the previous random variables. The parameters of the AR model are the same as those of [27].
- Artificial Neural Network (ANN) is a non-parametric learning model that simulates neuronal activity with a mathematical model and is based on mimicking the structure and function of neural networks in the brain. In this paper, the number of hidden layers of this ANN is one, and the spread of radial basis functions is set as 2000 according to the [51].
- Gray Model (GM) is based on gray system theory and can perform series forecasting, catastrophe forecasting, seasonal catastrophe forecasting, topological forecasting, and integrated system forecasting. We use the most commonly used gray forecasting model GM(1,1) model, which represents a differential equation model of order one and one variable. The GM(1,1) model group, the metabolic model, is the most desirable model.
- Kalman filter (KF), as one of the most popular filtering algorithms, is a filtering process based on linear equations to eliminate the effects of noise and other effects in the data, including the system, i.e., the process of making optimal predictions about the system. As suggested in [52], we set the initial state to  $[\frac{1}{n}, \dots, \frac{1}{n}]$ , where  $N$  is set as eight.
- Decision tree (DT) is an intuitive graphical analysis method for classification and regression problem solving, where different states are generated by each decision to form a decision tree, top-down from the root node to the leaf nodes to form different paths i.e., different solutions and to find the desired optimal path. In [53], the authors used the classification and regression tree (CART) to predict the traffic flow, which is robust to missing data.

In addition to the above common forecasting models, we also compared the traditional ELM, ELM optimized by Genetic Algorithm (GA), and ELM optimized by Gravitational Search Algorithm (GSA). The data recorded in Tables 3 and 4 are the average value of the data obtained by training 100 times under four datasets, A1, A2, A4, and A8.

**Table 3.** The MAPE(%) of different forecasting models under four datasets.

Models	Data A1	Data A2	Data A4	Data A8
SVR	14.34	12.22	12.23	12.47
AR	13.57	11.59	12.70	12.71
ANN	12.61	10.89	12.49	12.53
GM	12.49	10.90	13.22	12.89
KF	12.46	10.72	12.62	12.63
DT	12.08	10.86	12.34	13.62
ELM	11.92	10.32	12.09	12.58
GA-ELM	11.86	10.30	11.87	12.26
GSA-ELM	11.69	10.25	11.72	12.05
ABC-ELM	<b>11.40</b>	<b>9.95</b>	<b>11.26</b>	<b>11.90</b>

**Table 4.** The RMSE (vehs/h) of different forecasting models under four datasets.

Models	Data A1	Data A2	Data A4	Data A8
SVR	329.09	259.74	253.66	190.30
AR	301.44	214.22	226.12	166.71
ANN	299.64	212.95	225.86	166.50
GM	347.94	261.36	275.35	189.57
KF	322.03	239.87	250.51	187.48
DT	316.57	224.79	243.19	238.35
ELM	300.67	208.84	224.54	172.69
GA-ELM	291.42	211.43	228.57	169.25
GSA-ELM	287.89	203.04	221.39	163.24
ABC-ELM	<b>286.25</b>	<b>200.42</b>	<b>220.07</b>	<b>163.67</b>

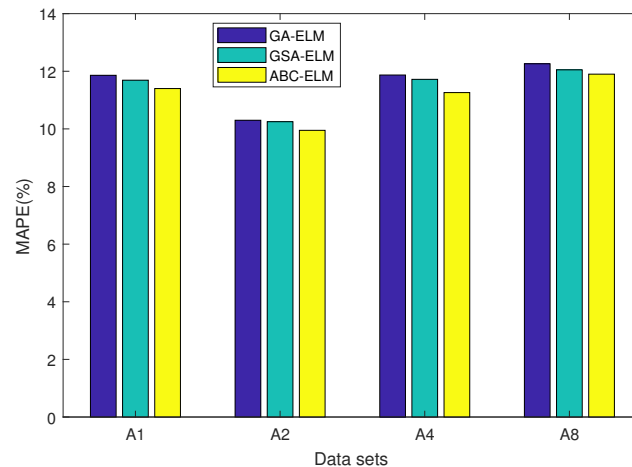
Tables 3 and 4 show the different two evaluation criteria, MAPE and RMSE scores, for each model with the same dataset, respectively. It can be seen that the ABC-ELM model achieves superior scores compared to the other models on all datasets.

In Table 3, ABC-ELM performs well compared to the other models in all four datasets. Compared with the traditional ELM model, the MAPE scores decreased by 4.3%, 3.5%, 6.9%, and 5.4% in the same four datasets A1, A2, A4, and A8, respectively. In Table 4, the RMSEs of the ABC-ELM model are 4.8%, 4.0%, 2.0%, and 5.2% lower than the RMSEs of the traditional ELM model at A1, A2, A4, and A8, respectively. It can be seen that the ABC-ELM model not only has the advantages of fast learning of the traditional ELM model but also has superior performance. In addition, compared with the ANN model, the MAPE scores decreased by 9.6%, 8.6%, 9.8%, and 5.0% in the same four datasets A1, A2, A4, and A8, respectively. In addition, the RMSEs of the ABC-ELM model are also 4.5%, 5.9%, 2.6%, and 1.7% lower than the RMSEs of the ANN model, respectively. In addition to the numerical advantages reflected in the evaluation criteria, the ANN model and the ABC-ELM model also include:

1. Since the connection weights of the input and implicit layers and the threshold of the implicit layer of ELM can be set randomly and do not need to be adjusted after setting, the connection weights  $\beta$  do not need to be adjusted iteratively. The above are determined once by solving a system of equations, which makes the learning speed of ELM significantly higher than that of ANN.
2. ANN, as a traditional gradient-based learning algorithm, also faces more complex problems, such as falling into local optimum, overfitting, etc. Overcoming these problems often requires some cost, while ELM does not need to overcome these problems, which leads to a much simpler network structure for ELM than ANN.
3. ELM can use non-differentiable functions as activation functions, but ANN is only applicable to differentiable functions.

We also compare traditional ELM and the improved optimization models based on the traditional ELM, including the GA-ELM model and GSA-ELM [15]. As shown in Figure 5,

it is clear that the ABC-ELM model outperforms the GSA-ELM in all four MAPE evaluation criteria with the same dataset, and ABC-ELM still matches the performance of GSA-ELM in the RMSE evaluation criteria. In addition, we also compared the Akaike information criterion (AIC) values of traditional ELM, GA-ELM, GSA-ELM, and ABC-ELM for the four datasets, which is a measure of the goodness of fit of statistical models. As shown in Table 5, the ABC-ELM model has lower AIC values compared to the GA-ELM model and GSA-ELM model for the same dataset. Therefore, we believe that the ABC-ELM has better forecasting accuracy.



**Figure 5.** Comparison of MAPE values of three learning models GA-ELM, GSA-ELM, and ABC-ELM. ABC-ELM has the lowest MAPE values in all four datasets A1, A2, A4, and A8.

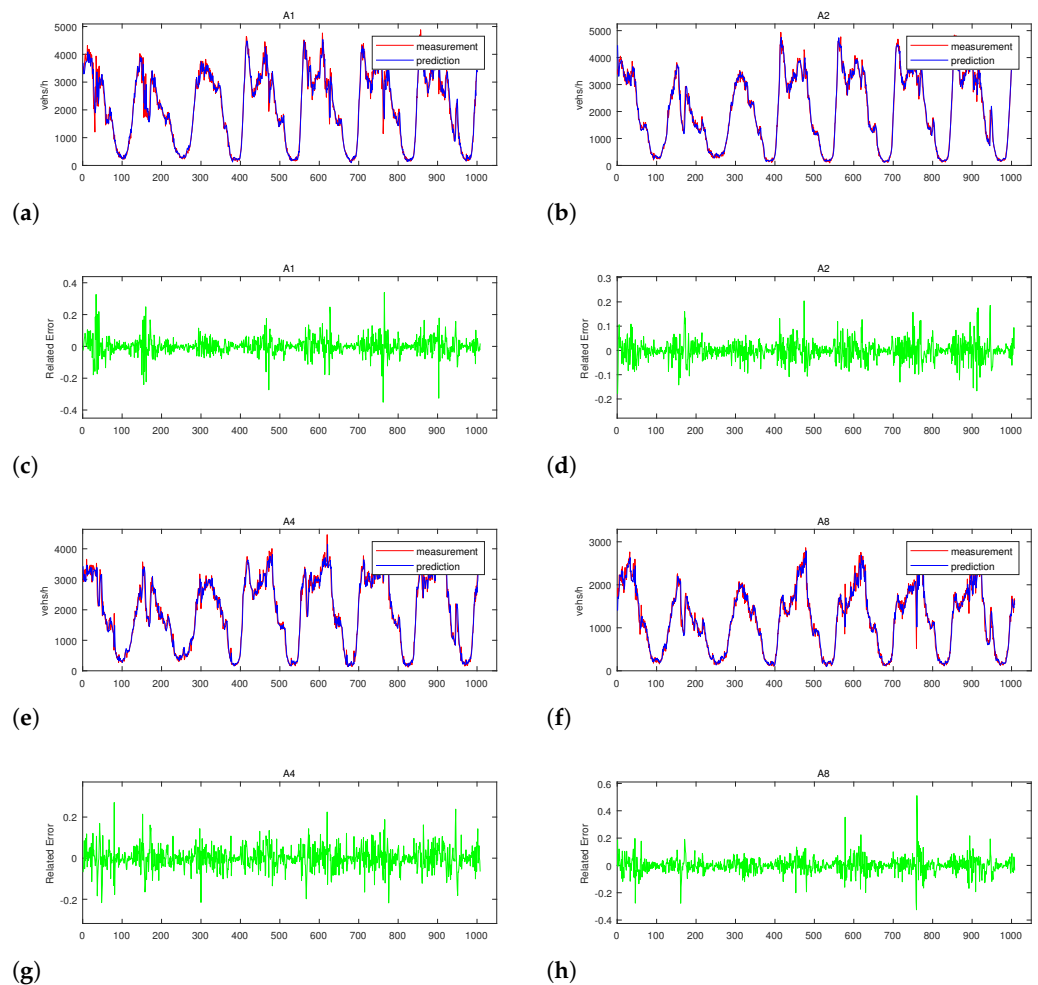
**Table 5.** The comparison of AIC for four models based on ELM.

Models	Data A1	Data A2	Data A4	Data A8
ELM	15.429	15.544	15.168	14.267
GA-ELM	15.416	15.522	15.136	14.161
GSA-ELM	15.324	15.431	15.066	14.151
<b>ABC-ELM</b>	<b>15.323</b>	<b>15.429</b>	<b>14.992</b>	<b>14.136</b>

Figure 6a–h represent the forecasting–measurement value comparison plots and the forecasting–measurement value relative error of the ABC-ELM model under four training sets A1, A2, A4, and A8, respectively. Where Figure 6a,b,e,f represent the forecasting–measurement value comparison plots of ABC-ELM, the blue line represents the forecasting, and the red line represents the measurement. As can be seen from the figure, the ABC-ELM model performs well in the traffic flow forecasting model, and the predicted results are almost consistent with the actual results. However, there are still shortcomings, especially when the traffic flow changes dramatically in a short period. Figure 6c,d,g,h represent the forecasting–measurement relative error of ABC-ELM. Its calculation formula can be expressed as:

$$\epsilon_i = m_i - p_i, \tag{11}$$

where  $\epsilon$  stands for relative error, and  $m$  and  $p$  represent the measurement and forecasting, respectively. It can be visually seen that the ABC-ELM model performance is good in four different scenarios, and the relative errors fluctuate above and below 0 in most cases. The relative error of our ABC-ELM model remains almost near 0 when the traffic flow is relatively stable, such as when the traffic flow is low and stable in the middle of the night or when the traffic flow gradually increases in the early morning. However, when the traffic volume changes rapidly, the relative error of the ABC-ELM model also fluctuates greatly.

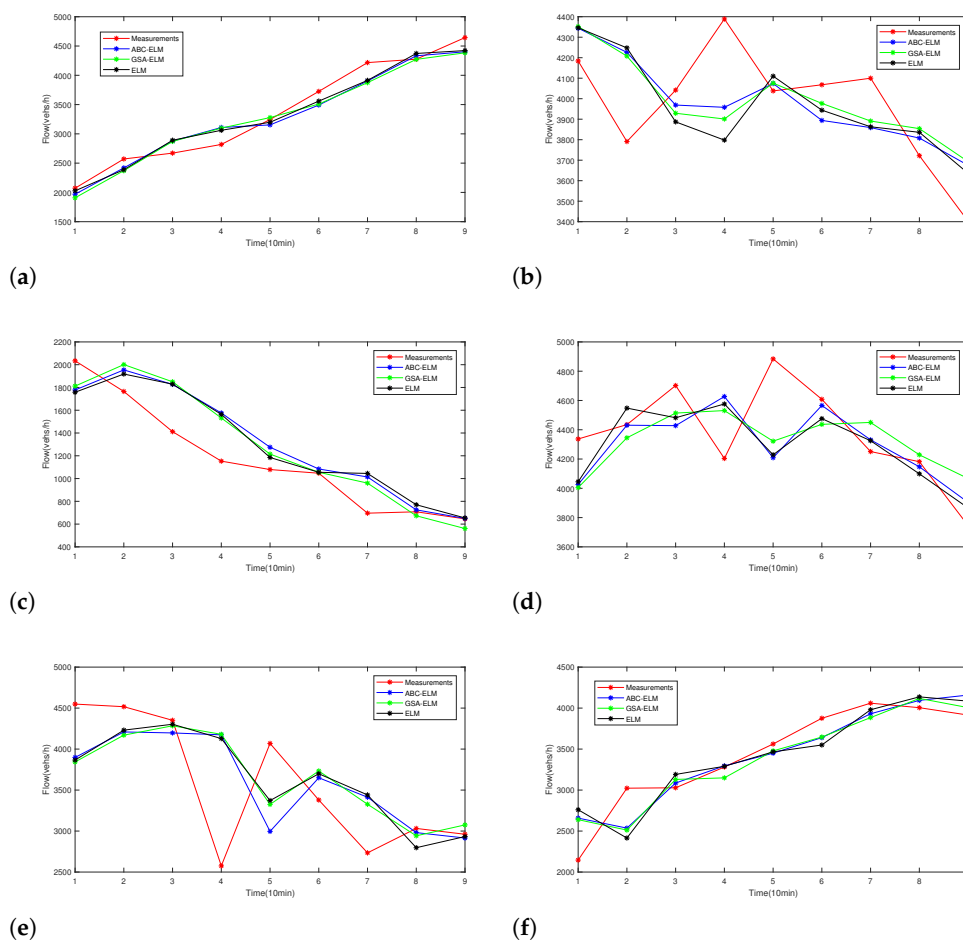


**Figure 6.** The (a–h) represent the forecasting–measurement value comparison plots and the forecasting–measurement value relative error of ABC-ELM model under four training sets.

Finally, Figure 7a–f represent the partial comparison of predicted values for each learning model in different cases. Since we are taking nine elements as input feature values, the predicted value segments obtained here are also nine moments. It represents the traffic flow values for the first ten-minute moment, the traffic flow values for the second ten-minute moment, and so on. The red line represents the actual value, the blue line represents the predicted value from the ABC-ELM model, the green line represents the predicted value from the GSA-ELM model, and the black line represents the predicted value from the ELM model.

The period selected for part Figure 7a is early morning, and it can be seen that the traffic flow on the road shows a steady increase with time. The predicted values of ABC-ELM, GSA-ELM, and ELM models also show a steady increase, but the ABC-ELM model is smoother, and it is smoother than the ELM model. In addition, the ABC-ELM model is more sensitive to changes in traffic flow values.

The period selected for part Figure 7b is around 11:40 a.m., which is at the peak of the traffic flow, and the traffic flow is large and changes repeatedly within a short period. It can be seen that the ABC-ELM model forecasting at the fourth moment is the closest to the actual traffic flow. Although there is still a certain gap with the actual value error, the ABC-ELM model is more accurate compared with the same type of model.



**Figure 7.** The part of (a–f) shows the performance of the three models under six different classical periods, respectively, where the classical periods include realistic scenarios, such as steady increases and sharp fluctuations in values.

Figure 7c is selected with the period around 10:00 p.m. As night comes, the traffic flow gradually decreases, and the three models perform well.

Figure 7d is also taken from the peak traffic flow hours. The GSA-ELM model is not as responsive as the ABC-ELM model to large changes in traffic flow in the short-term. In addition, most of the time the gradient of the predicted values of the GSA-ELM model is insufficient, When the actual traffic flow changes significantly in the short-term, we have less accurate forecasts than with the ABC-ELM model.

Figure 7e shows the predictions of each model when a huge drop in traffic flow occurs. It can be seen that during the fourth moment, a precipitous drop in the actual traffic flow occurs, followed by an increase to about 4000. While the ABC-ELM model deviates more from the actual value in the fifth moment of forecasting, this is because the ABC-ELM model is more sensitive to changes in the data.

Figure 7f shows a steady growth trend similar to a convex function, and the ABC-ELM model works well for that case as well.

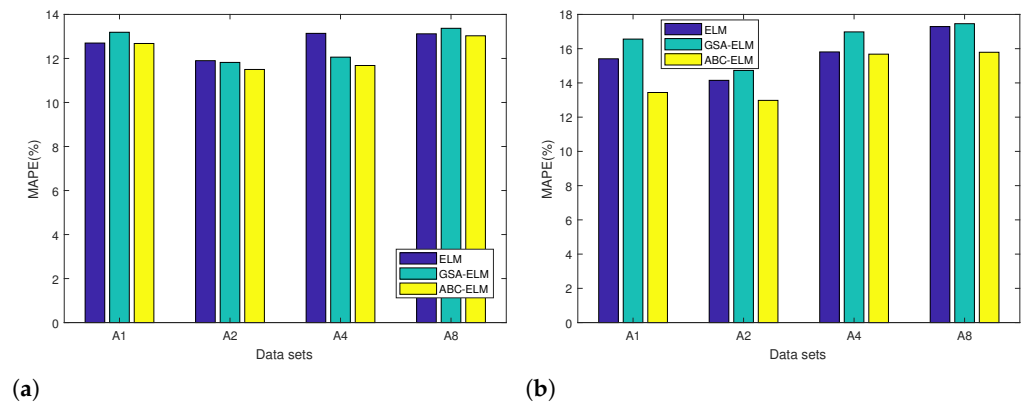
### 3.5. Ablation Study

In this subsection, we focus on the performance of the ABC-ELM model with different aggregated data. The subject of a good short-term traffic flow forecast is not to unilaterally achieve faster and more accurate forecasts, but to achieve advanced forecasts. The dataset previously used was aggregated data every ten minutes; we also compare the performance

of the model with a dataset aggregated every five minutes and a dataset aggregated every twenty minutes.

We used the ABC-ELM model with the same model parameters as mentioned above, as shown in Tables 1 and 2, and tested it in datasets with aggregation intervals of 5 min and 20 min, respectively. As shown in Table 6, the MAPE value of the ABC-ELM model achieves the lowest value for an aggregation interval of 10 min. This indicates that the ABC-ELM model performs better than the other two aggregation intervals when aggregating data every 10 min. In addition, the ABC-ELM model outperformed every 5 min aggregation interval at every 20 min aggregation interval.

Moreover, we compared the performance of the three models ELM, GSA-ELM, and ABC-ELM for datasets with an aggregation interval of five minutes and an aggregation interval of ten minutes. Figure 8a,b show the MAPE values of the three models ELM, GSA-ELM, and ABC-ELM for the dataset with an aggregation interval of 5 min and aggregation interval of 20 min, respectively. Overall, ABC-ELM also shows better results than the other two models in more dense or sparse traffic flow prediction problems.



**Figure 8.** Comparison of MAPE (a,b) values of three learning models ELM, GSA-ELM, and ABC-ELM at 5 min and 20 min aggregation interval. ABC-ELM has the lowest MAPE values in all four datasets A1, A2, A4, A8.

**Table 6.** The MAPE(%) of ABC-ELM at different aggregation intervals.

Aggregation Intervals	Data A1	Data A2	Data A4	Data A8
5 min	12.68	11.50	11.68	13.03
10 min	<b>11.40</b>	<b>9.95</b>	<b>11.26</b>	<b>11.90</b>
20 min	13.44	12.98	15.68	15.79

### 3.6. Discussion

The purpose of this paper is to propose a more accurate and efficient forecasting model to address the problems of existing short-term traffic flow forecasting methods. From the results of the experiments, our proposed hybrid model achieved the expected results. In the results of comparing each model using experimental evaluation criteria, our proposed ABC-ELM model achieved more accurate predictions. First, we attribute all this mainly to the fact that the hybrid model unlocks the potential of the forecasting model using the features of the optimization algorithm. Since the parameters of ELM are determined randomly, this indicates a huge optimization space for the learning process of ELM. We unlocked the potential of ELM using the ABC optimization algorithm. Second, the experiment shows that GA-ELM, GSA-ELM, and ABC-ELM all use optimization algorithms to optimize ELM, but the ABC-ELM achieved higher prediction accuracy. This is mainly due to the fact that the ABC algorithm uses fewer control parameters but maintains a certain level of robustness. It also finds the current local solution and the global optimal solution just by comparing the advantages and disadvantages of the problem in each round

of iterations. So, the probability of finding the optimal solution increases significantly. However, there are still shortcomings in this paper. First, the experiments designed in this paper are more applicable to ABC-ELM models with a priori defined variables and implemented data aggregation. Second, our proposed ABC-ELM model and the experimental design in the paper do not use preprocessing of irregular time series. This leads to traditional methods used to process time series data introducing bias and failing to model the temporal irregularities of incomplete time series. We consider the two irregular time series preprocessing methods proposed in [54] to be constructive. In the future, we will introduce the irregular time series preprocessing methods, and the idea of hybrid models can be applied to many regression problems, such as predicting spatial CO<sub>2</sub> concentration, forecasting airport passenger traffic, etc.

#### 4. Conclusions

In this paper, we discuss the common forecasting models that currently exist in the field of short-term traffic flow forecasting and their advantages and disadvantages. We consider the development potential of traditional machine learning models proposing the use of data-driven formal optimization models. We exemplify a hybrid learning model for short-term traffic flow forecasting that learns nonlinear features in traffic flow sequences by using an ELM and an ABC algorithm to optimize the parameter values in the ELM. Through the experimental design, we put the commonly used short-term traffic flow prediction models under the same dataset and optimal model parameters for prediction experiments. According to the experimental comparison results, the prediction accuracy of our proposed learning model on all four benchmark datasets exceeds that of the most mainstream prediction models currently available. Moreover, we also compared the GA-ELM and GSA-ELM models that also use the hybrid model concept. In addition, we also added the AIC evaluation metric, and the results show that the prediction accuracy of the ABC-ELM model is higher than the other two. The main benefit is that the ABC algorithm only needs to compare the advantages and disadvantages of the problem during the iterative process, and the global optimum is finally made to emerge in the population through the local optimization-seeking behavior of each individual worker bee. This work verifies the feasibility of the hybrid model concept, and in the future, we hope to apply this concept to more regression problems and even to problems such as classification.

**Author Contributions:** Conceptualization, X.Y. and T.Z.; methodology, L.L. and X.L.; software, L.L.; validation, B.H. and H.D.; formal analysis, L.L.; investigation, L.L.; data curation, L.L.; writing—original draft preparation, L.L.; writing—review and editing, L.L.; visualization, L.L.; supervision, T.Z.; project administration, L.L. and X.L.; funding acquisition, T.Z.; T.Z., L.L. and X.L. contribute equally. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the National Natural Science Foundation of China (No. 61902232), the 2021 Guangdong Basic and Applied Basic Research Regional Joint Foundation (No. 2021A1515110716), Guangzhou Scientific and Technological Plan Project (No. 202102021098), the 2022 Guangdong Basic and Applied Basic Research Foundation (No. 2022A1515011590), the STU Incubation Project for the Research of Digital Humanities and New Liberal Arts (No. 2021DH-3), the 2020 Li Ka Shing Foundation Cross-Disciplinary Research Grant (No. 2020LKSFG05D), and the Open Fund of Guangdong Provincial Key Laboratory of Infectious Diseases and Molecular Immunopathology (GDKL202212).

**Data Availability Statement:** The data that support the findings of this study are available from the corresponding author upon reasonable request.

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

#### References

1. Zhou, T.; Han, G.; Xu, X.; Han, C.; Huang, Y.; Qin, J. A learning-based multimodel integrated framework for Dynamic traffic flow forecasting. *Neural Process. Lett.* **2019**, *49*, 407–430. [[CrossRef](#)]
2. Cai, L.; Yu, Y.; Zhang, S.; Song, Y.; Xiong, Z.; Zhou, T. A Sample-rebalanced Outlier-rejected k-nearest Neighbour Regression Model for Short-Term Traffic Flow Forecasting. *IEEE Access* **2020**, *8*, 22686–22696. [[CrossRef](#)]



3. Zhou, T.; Jiang, D.; Lin, Z.; Han, G.; Xu, X.; Qin, J. Hybrid dual Kalman filtering model for short-term traffic flow forecasting. *Intell. Transp. Syst.* **2019**, *13*, 1023–1032. [[CrossRef](#)]
4. Pei, M.; Lin, P.; Liu, R.; Ma, Y. Flexible transit routing model considering passengers' willingness to pay. *Intell. Transp. Syst.* **2019**, *13*, 841–850. [[CrossRef](#)]
5. Li, S.; Lyu, D.; Huang, G.; Zhang, X.; Gao, F.; Chen, Y.; Liu, X. Spatially varying impacts of built environment factors on rail transit ridership at station level: A case study in Guangzhou, China. *J. Transp. Geogr.* **2020**, *82*, 102631. [[CrossRef](#)]
6. Li, S.; Zhuang, C.; Tan, Z.; Gao, F.; Lai, Z.; Wu, Z. Inferring the trip purposes and uncovering spatio-temporal activity patterns from dockless shared bike dataset in Shenzhen, China. *J. Transp. Geogr.* **2021**, *91*, 102974. [[CrossRef](#)]
7. Cai, L.; Lei, M.; Zhang, S.; Yu, Y.; Zhou, T.; Qin, J. A noise-immune LSTM network for short-term traffic flow forecasting. *Chaos* **2020**, *30*, 023135. [[CrossRef](#)]
8. Qin, J.; He, Z.S. A SVM face recognition method based on Gabor-featured key points. In Proceedings of the 2005 International Conference on Machine Learning and Cybernetics, Guangzhou, China, 18–21 August 2005; Volume 8, pp. 5144–5149.
9. Polikar, R. Ensemble learning. In *Ensemble Machine Learning*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 1–34.
10. Cui, Z.; Huang, B.; Dou, H.; Cheng, Y.; Guan, J.; Zhou, T. A Two-Stage Hybrid Extreme Learning Model for Short-term Traffic Flow Forecasting. *Mathematics* **2022**, *10*, 2087. [[CrossRef](#)]
11. Wei, Y.; Zheng, S.; Yang, X.; Huang, B.; Tan, G.; Zhou, T. A noise-immune extreme learning machine for short-term traffic flow forecasting. In Proceedings of the International Conference on Smart Transportation and City Engineering, Chongqing, China, 6–8 August 2021; Volume 12050, pp. 599–604.
12. Cai, L.; Chen, Q.; Cai, W.; Xu, X.; Zhou, T.; Qin, J. SVR-GSA: A hybrid learning based model for short-term traffic flow forecasting. *Intell. Transp. Syst.* **2019**, *13*, 1348–1355. [[CrossRef](#)]
13. Cai, W.; Yang, J.; Yu, Y.; Song, Y.; Zhou, T.; Qin, J. PSO-ELM: A Hybrid Learning Model for Short-term Traffic Flow Forecasting. *IEEE Access* **2020**, *8*, 6505–6514. [[CrossRef](#)]
14. Zheng, S.; Zhang, S.; Song, Y.; Lin, Z.; Dazhi, J.; Zhou, T. A noise-immune boosting framework for short-term traffic flow forecasting. *Complexity* **2021**. [[CrossRef](#)]
15. Cui, Z.; Huang, B.; Dou, H.; Tan, G.; Zheng, S.; Zhou, T. GSA-ELM: A Hybrid Learning Model for Short-Term Traffic Flow Forecasting. *Intell. Transp. Syst.* **2022**, *16*. [[CrossRef](#)]
16. Box, G.E.; Jenkins, G.M.; Reinsel, G.C.; Ljung, G.M. *Time Series Analysis: Forecasting and Control*; John Wiley & Sons: Hoboken, NJ, USA, 2015.
17. Huang, D.R.; Song, J.; Wang, D.C.; Cao, J.Q.; Li, W. Forecasting model of traffic flow based on ARMA and wavelet transform. *Comput. Eng. Appl.* **2006**, *42*, 191–194.
18. Han, C.; Song, S.; Wang, C.h. A real-time short-term traffic flow adaptive forecasting method based on ARIMA model. *Acta Simulata Syst. Sin.* **2004**, *7*, 3.
19. Zhang, S.; Song, Y.; Jiang, D.; Zhou, T.; Qin, J. Noise-identified Kalman filter for short-term traffic flow forecasting. In Proceedings of the 2019 15th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN), Shenzhen, China, 11–13 December 2019; pp. 462–466. [[CrossRef](#)]
20. Cai, L.; Zhang, Z.; Yang, J.; Yu, Y.; Zhou, T.; Qin, J. A noise-immune Kalman filter for short-term traffic flow forecasting. *Phys. A Stat. Mech. Its Appl.* **2019**, *536*, 122601. [[CrossRef](#)]
21. Shi, G.; Guo, J.; Huang, W.; Williams, B.M. Modeling seasonal heteroscedasticity in vehicular traffic condition series using a seasonal adjustment approach. *J. Transp. Eng.* **2014**, *140*, 04014012. [[CrossRef](#)]
22. Williams, B.M.; Hoel, L.A. Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results. *J. Transp. Eng.* **2003**, *129*, 664–672. [[CrossRef](#)]
23. Li, L.; Qin, L.; Qu, X.; Zhang, J.; Wang, Y.; Ran, B. Day-ahead traffic flow forecasting based on a deep belief network optimized by the multi-objective particle swarm algorithm. *Knowl.-Based Syst.* **2019**, *172*, 1–14. [[CrossRef](#)]
24. Zhang, Y.; Ye, Z. Short-term traffic flow forecasting using fuzzy logic system methods. *J. Intell. Transp. Syst.* **2008**, *12*, 102–112. [[CrossRef](#)]
25. Arqub, O.A.; Al-Smadi, M.; Momani, S.; Hayat, T. Application of reproducing kernel algorithm for solving second-order, two-point fuzzy boundary value problems. *Soft Comput.* **2017**, *21*, 7191–7206. [[CrossRef](#)]
26. Li, L.; Qu, X.; Zhang, J.; Wang, Y.; Ran, B. Traffic speed prediction for intelligent transportation system based on a deep feature fusion model. *J. Intell. Transp. Syst.* **2019**, *23*, 605–616. [[CrossRef](#)]
27. Zhou, T.; Han, G.; Xu, X.; Lin, Z.; Han, C.; Huang, Y.; Qin, J.  $\delta$ -agree AdaBoost stacked autoencoder for short-term traffic flow forecasting. *Neurocomputing* **2017**, *247*, 31–38. [[CrossRef](#)]
28. Lu, H.; Huang, D.; Youyi, S.; Jiang, D.; Zhou, T.; Qin, J. ST-TrafficNet: A Spatial-Temporal Deep Learning Network for Traffic Forecasting. *Electronics* **2020**, *9*, 1–474. [[CrossRef](#)]
29. Lu, H.; Ge, Z.; Song, Y.; Jiang, D.; Zhou, T.; Qin, J. A temporal-aware lstm enhanced by loss-switch mechanism for traffic flow forecasting. *Neurocomputing* **2021**, *427*, 169–178. [[CrossRef](#)]
30. Fang, W.; Zhuo, W.; Yan, J.; Song, Y.; Jiang, D.; Zhou, T. Attention Meets Long Short-term Memory: A Deep Learning Network for Traffic Flow Forecasting. *Phys. A Stat. Mech. Its Appl.* **2022**, *587*, 126485. [[CrossRef](#)]
31. Fang, W.; Zhuo, W.; Song, Y.; Yan, J.; Zhou, T.; Qin, J.  $\Delta_{free}$ -LSTM: An Error Distribution Free Deep Learning for Short-term Traffic Flow Forecasting. *Neurocomputing* **2023**.

32. Huang, B.; Dou, H.; Luo, Y.; Li, J.; Wang, J.; Zhou, T. Adaptive Spatiotemporal Transformer Graph Network for Traffic Flow Forecasting by IoT Loop Detectors. *IEEE Internet Things J.* **2022**. [[CrossRef](#)]
33. Li, H.; Yang, S.; Luo, Y.; Li, J.; Song, Y.; Zhou, T. Spatial Dynamic Graph Convolutional Network for Traffic Flow Forecasting. *Appl. Intell.* **2022**. [[CrossRef](#)]
34. Yang, S.; Li, H.; Luo, Y.; Li, J.; Song, Y.; Zhou, T. Spatiotemporal Adaptive Fusion Graph Network for Short-Term Traffic Flow Forecasting. *Mathematics* **2022**, *10*, 1594. [[CrossRef](#)]
35. Zhou, T.; Dou, H.; Tan, J.; Song, Y.; Wang, F.; Wang, J. Small dataset solves big problem: An outlier-insensitive binary classifier for inhibitory potency prediction. *Knowl.-Based Syst.* **2022**. [[CrossRef](#)]
36. Dou, H.; Tan, J.; Wei, H.; Wang, F.; Yang, J.; Ma, X.G.; Wang, J.; Zhou, T. Transfer inhibitory potency prediction to binary classification: A model only needs a small training set. *Comput. Methods Progr. Biomed.* **2022**. [[CrossRef](#)] [[PubMed](#)]
37. Huang, G.B.; Zhu, Q.Y.; Siew, C.K. Extreme learning machine: A new learning scheme of feedforward neural networks. In Proceedings of the 2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No. 04CH37541), Budapest, Hungary, 25–29 July 2004; Volume 2, pp. 985–990.
38. Ahila, R.; Sadasivam, V.; Manimala, K. An integrated PSO for parameter determination and feature selection of ELM and its application in classification of power system disturbances. *Appl. Soft Comput.* **2015**, *32*, 23–37. [[CrossRef](#)]
39. Yuan, Y.; Quan, T.; Song, Y.; Guan, J.; Zhou, T.; Wu, R. Noise-immune Extreme Ensemble Learning for Early Diagnosis of Neuropsychiatric Systemic Lupus Erythematosus. *IEEE J. Biomed. Health Inform.* **2022**. [[CrossRef](#)]
40. Ozturk, C.; Karaboga, D. Hybrid artificial bee colony algorithm for neural network training. In Proceedings of the 2011 IEEE Congress of Evolutionary Computation (CEC), New Orleans, LA, USA, 5–8 June 2011; pp. 84–88.
41. Huang, G.B.; Zhou, H.; Ding, X.; Zhang, R. Extreme learning machine for regression and multiclass classification. *IEEE Trans. Syst. Man, Cybern. Part B* **2011**, *42*, 513–529. [[CrossRef](#)] [[PubMed](#)]
42. Akay, B.; Karaboga, D. A modified artificial bee colony algorithm for real-parameter optimization. *Inf. Sci.* **2012**, *192*, 120–142. [[CrossRef](#)]
43. Karaboga, D.; Akay, B. A comparative study of artificial bee colony algorithm. *Appl. Math. Comput.* **2009**, *214*, 108–132. [[CrossRef](#)]
44. Raza, M.Q.; Khosravi, A. A review on artificial intelligence based load demand forecasting techniques for smart grid and buildings. *Renew. Sustain. Energy Rev.* **2015**, *50*, 1352–1372. [[CrossRef](#)]
45. Huang, G.B.; Zhu, Q.Y.; Siew, C.K. Extreme learning machine: Theory and applications. *Neurocomputing* **2006**, *70*, 489–501. [[CrossRef](#)]
46. Mahmood, S.F.; Marhaban, M.H.; Rokhani, F.Z.; Samsudin, K.; Arigbabu, O.A. FASTA-ELM: A fast adaptive shrinkage/thresholding algorithm for extreme learning machine and its application to gender recognition. *Neurocomputing* **2017**, *219*, 312–322. [[CrossRef](#)]
47. Feng, X.; Ling, X.; Zheng, H.; Chen, Z.; Xu, Y. Adaptive multi-kernel SVM with spatial-temporal correlation for short-term traffic flow prediction. *IEEE Trans. Intell. Transp. Syst.* **2018**, *20*, 2001–2013. [[CrossRef](#)]
48. Wang, Y.; Van Schuppen, J.H.; Vrancken, J. Prediction of traffic flow at the boundary of a motorway network. *IEEE Trans. Intell. Transp. Syst.* **2013**, *15*, 214–227. [[CrossRef](#)]
49. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [[CrossRef](#)]
50. Hsieh, W.W. *Machine Learning Methods in the Environmental Sciences: Neural Networks and Kernels*; Cambridge University Press: Cambridge, UK, 2009.
51. Zhu, J.Z.; Cao, J.X.; Zhu, Y. Traffic volume forecasting based on radial basis function neural network with the consideration of traffic flows at the adjacent intersections. *Transp. Res. Part Emerg. Technol.* **2014**, *47*, 139–154. [[CrossRef](#)]
52. Gallo, C.; Conto, F.; Fiore, M. A neural network model for forecasting CO<sub>2</sub> emission. *Agris-Line Pap. Econ. Inform.* **2014**, *6*, 31–36.
53. Xu, Y.; Kong, Q.J.; Liu, Y. Short-term traffic volume prediction using classification and regression trees. In Proceedings of the 2013 IEEE Intelligent Vehicles Symposium (IV), Gold Coast, Australia, 23–26 June 2013; pp. 493–498.
54. Weerakody, P.B.; Wong, K.W.; Wang, G.; Ela, W. A review of irregular time series data handling with gated recurrent neural networks. *Neurocomputing* **2021**, *441*, 161–178. [[CrossRef](#)]