applied sciences

MDPI

Article

Vessel Trajectory Prediction Model Based on AIS Sensor Data and Adaptive Chaos Differential Evolution Support Vector Regression (ACDE-SVR)

Jiao Liu, Guoyou Shi * and Kaige Zhu

Key Laboratory of Navigation Safety Guarantee of Liaoning Province, Navigation College, Dalian Maritime University, Dalian 116026, China

* Correspondence: sgydmu@dlmu.edu.cn; Tel.: +86-180-1890-1579

Received: 5 June 2019; Accepted: 22 July 2019; Published: 25 July 2019

Featured Application: Authors are encouraged to provide a concise description of the specific application or a potential application of the work. This section is not mandatory.

Abstract: There are difficulties in obtaining accurate modeling of ship trajectories with traditional prediction methods. For example, neural networks are prone to falling into local optima and there are a small number of Automatic Identification System (AIS) information samples regarding target ships acquired in real time at sea. In order to improve the accuracy of ship trajectory predictions and solve these problems, a trajectory prediction model based on support vector regression (SVR) is proposed. Ship speed, course, time stamp, longitude and latitude from AIS data were selected as sample features and the wavelet threshold de-noising method was used to process the ship position data. The adaptive chaos differential evolution (ACDE) algorithm was used to optimize the internal model parameters to improve convergence speed and prediction accuracy. AIS sensor data corresponding to a certain section of the Tianjin Port ships were selected, on which SVR, Recurrent Neural Network (RNN) and Back Propagation (BP) neural network model trajectory prediction simulations were carried out. A comparison of the results shows that the trajectory prediction model based on ACDE-SVR has higher and more stable prediction accuracy, requires less time and is simple, feasible and efficient.

Keywords: vessel trajectory prediction; AIS sensor data; support vector regression (SVR); adaptive chaos differential evolution algorithm (ACDE)

1. Introduction

With the rapid development of economic globalization, shipping has become the main form of international trade. As an important means of transportation at sea, ships have made tremendous contributions to economic development and social progress. The large-scale and diversified development of ships has also brought great security risks to maritime navigation. According to statistics from relevant departments [1], in recent years, more than 200 ships have been involved in shipwrecks at sea each year and more than half of the losses were caused by collisions. Investigation [1] of these marine collision accidents has found that the actual situations encountered during maritime navigation are often highly complicated. Especially in multi-ship collision avoidance, the ambiguity of a target ship's intentions and inability to make real-time and effective collision avoidance decisions are two of the largest factors contributing to the frequent occurrence of collision accidents at sea. Considering these factors, collision avoidance decisions cannot be based only on the current navigational information but should instead be integrated into the target ship's future navigational behavior for a certain period of time. This allows for collision avoidance decisions to be made ahead of time to

some extent, which effectively improves the reliability of the decisions and reduces the collision risk. Therefore, a ship trajectory prediction model with high precision and real-time prediction capability is an urgent necessity.

Current trajectory prediction methods usually adopt the traditional Markov model [2–5], particle filter algorithm [6,7], simulated annealing algorithm [8] and Kalman filter algorithm [9]. These methods often have the following shortcomings: First, the ship kinematic equations must be established and consideration of hydrological environmental factors such as wind and current greatly increases the modeling complexity and difficulty; Second, according to the needs of marine collision avoidance decision-making, trajectory prediction must often occur in real time. It is often difficult to establish real-time and accurate mathematical models, as many are only suitable for ideal research. In contrast to traditional modeling methods, with the rapid rise of the artificial intelligence era, more machine learning algorithms are being gradually applied to trajectory prediction, among which back propagation (BP) neural networks are common [10,11]. However, these algorithms fall easily into local optima and a problem often arises where the amount of data is too small for accurate BP neural network training [12]. As a result, the accuracy is insufficient for real-time data training and modeling at sea.

Considering these limitations, a vessel trajectory prediction model based on adaptive chaos differential evolution algorithm [13] support vector regression (ACDE-SVR) is proposed herein. The SVR model ensures high prediction accuracy despite the limited samples; this solve the problem of insufficient accuracy of models such as neural network by the real-time acquisition of AIS data for track prediction at sea. The prediction accuracy of the support vector regression (SVR) is closely related to the selection of parameters in the model. To further improve the prediction accuracy, the heuristic algorithm is used to optimize the parameters of the SVR model. The improved differential evolution (DE) algorithm-ACDE algorithm is used for parameter optimization. The speed, course, ship position and Unix time from AIS information are used as model sample characteristics and a multi-dimensional trajectory time series during maritime navigation is successfully predicted. The results provide a better understanding of ship dynamics and contribute to improved determination of collision avoidance decisions.

2. Related Concepts

2.1. SVM System and SVR

SVM (Support Vector Machine) is a novel supervised learning algorithm that was first proposed by Cortes and Vapnik [14] in 1995. It is based on the statistical theory concept of the VC (Vapnik Chervonenkis) dimension and can improve the generalization performance of learning machines by seeking structural risk minimization [15]. The VC-dimension of the set of indicator functions $f(x,a), a \in \Lambda$, is the maximum number of a set of vectors $x_1, x_2, x_3, ..., x_h$ that can be separated into two classes in all 2^h possible ways using functions of the set. If for any n there exists a set of n vectors that can be shattered by the set of the set $f(x,a), a \in \Lambda$, then the VC dimension is equal to infinity [16]. VC dimension measures the capacity of a hypothesis space. Capacity is a measure of complexity and measures the expressive power, richness or flexibility of a set of functions by assessing how wiggly its members can be. Notably, SVM can obtain good statistical laws under the premise of limited samples [17]. SVM is divided into support vector classification (SVC) and SVR, which are used to solve data classification and data regression problems, respectively. This paper uses an SVR model for vessel trajectory prediction.

Given a set of sample data $\{(\boldsymbol{x}_i, y_i) | \, i = 1, 2, ..., n\}$ for prediction, where $\boldsymbol{x}_i \in R^n$ are the input variables and $y_i \in R$ are the output variables. The core of the SVR model is the kernel function. By introducing a kernel function, SVM subtly solves the inner product operation in high-dimensional feature space, which solves the problem of nonlinear classification and prediction. Commonly used kernel functions mainly include linear kernel functions, polynomial kernel functions, Radial Basis Function (RBF) and sigmoid kernel functions. Among them, RBF is the most widely used kernel

function, which can realize nonlinear mapping [18,19]. It has better performance for both large samples and small samples and has fewer parameters than polynomial kernel functions. In this study, RBF function is employed as the kernel function for the SVR model [20]:

$$K(\boldsymbol{x}, \boldsymbol{x}_i) = \varphi(\boldsymbol{x}) \cdot \varphi(\boldsymbol{x}_i) = \exp\left(-g\|\boldsymbol{x} - \boldsymbol{x}_i\|^2\right), \tag{1}$$

where K stands for kernel function; $\boldsymbol{x}$ and $\boldsymbol{x}_i$ are the vectors in initial low-dimensional feature space and g is the kernel function parameter; $\varphi(\boldsymbol{x}_i)$ is the nonlinear mapping function, which can map the input data into the high dimensional feature space; The operator $\cdot$ is the inner symbol. Through RBF kernel function mapping, each sample point (x_i, y_i) is fitted as closely as possible to the following SVR linear model:

$$f(\boldsymbol{x}) = \boldsymbol{\omega}^T \boldsymbol{x} + b, \tag{2}$$

where $\boldsymbol{\omega}$ is the weight vector, with $\boldsymbol{\omega} \in R^n$; and b is the bias, with $b \in R$.

SVR takes an ε-insensitive function as a loss function. First, a constant ε is defined for a specific problem, where $\varepsilon > 0$. For a sample $(\boldsymbol{x}_i, y_i)$, if $\left|y_i - \boldsymbol{\omega}^T \varphi(\boldsymbol{x}_i) - b\right| \le \varepsilon$, there is no loss; otherwise, the corresponding loss is $\left|y_i - \boldsymbol{\omega}^T \varphi(\boldsymbol{x}_i) - b\right| - \varepsilon$. That is, the ε-insensitive loss function can be written as:

$$err(\boldsymbol{x}_i, y_i) = \begin{cases} 0 & \left|y_i - \boldsymbol{\omega}^T \varphi(\boldsymbol{x}_i) - b\right| \le \varepsilon \\ \left|y_i - \boldsymbol{\omega}^T \varphi(\boldsymbol{x}_i) - b\right| - \varepsilon & \left|y_i - \boldsymbol{\omega}^T \varphi(\boldsymbol{x}_i) - b\right| > \varepsilon \end{cases}. \tag{3}$$

As shown in Figure 1, an interval band with $f(\boldsymbol{x})$ as the center and a width of 2ε is established. If the training sample falls into the interval band, the prediction is considered correct.

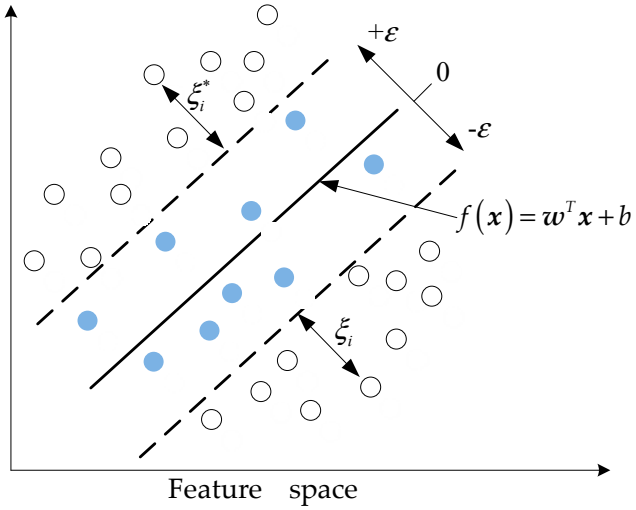


Figure 1. SVR schematic diagram. The area between the two dashed lines represents the ε-interval band; the predicted loss of samples falling into this area is not calculated.

In order to minimize the loss function, based on the criterion of structural risk minimization [21], determining the constraints $\boldsymbol{\omega}$ and b and optimizing the objective function, the following optimization criterion is constructed:

$$\min_{\boldsymbol{\omega}, b, \xi, \xi^*} \frac{1}{2}\boldsymbol{\omega}^T \boldsymbol{\omega} + C\sum_{i=1}^{n}\left(\xi_i + \xi_i^*\right), \tag{4}$$

$$\text{s.t.} \quad \begin{aligned} & y_i - \left(\boldsymbol{\omega}^T \boldsymbol{x}_i + b\right) \le \varepsilon + \xi_i \\ & \left(\boldsymbol{\omega}^T \boldsymbol{x}_i + b\right) - y_i \le \varepsilon + \xi_i^* \\ & \xi_i, \xi_i^* \ge 0 \end{aligned} \tag{5}$$

This is the convex quadratic programming problem of the original problem of SVR, where C is the penalty factor and $C > 0$, which aims to find a compromise between generalization performance and training error; ε is the maximum tolerance value; and ξ_i and ξ_i^* are slack variables to avoid over-training. Combining these variables allows the model a certain degree of fault tolerance [22]. The Lagrange function can be obtained by introducing Lagrange multipliers:

$$\begin{aligned} &L\left(\boldsymbol{\omega}, b, \alpha_i, \alpha_i^*, \xi_i, \xi_i^*, \mu_i, \mu_i^*\right) \\ &= \tfrac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + C\sum_{i=1}^{n}\left(\xi_i + \xi_i^*\right) - \sum_{i=1}^{n}\mu_i\xi_i - \sum_{i=1}^{n}\mu_i^*\xi_i^* \\ &+ \sum_{i=1}^{n}\alpha_i\left(\boldsymbol{\omega}^T\varphi(\boldsymbol{x}_i) + b - y_i - \varepsilon - \xi_i\right) + \sum_{i=1}^{n}\alpha_i^*\left(y_i - \boldsymbol{\omega}^T\varphi(\boldsymbol{x}_i) - b - \varepsilon - \xi_i^*\right) \end{aligned} \tag{6}$$

where $\alpha_i^{(*)}$ and $\mu_i^{(*)}$ are Lagrange multipliers and $0 \le \alpha_i^{(*)}, \mu_i^{(*)} \le C$.

The partial derivatives of $\boldsymbol{\omega}, b, \alpha_i$ and e_i are solved separately. Letting the partial derivatives be equal to zero [23] yields:

$$\begin{aligned} \omega &= \sum_{i=1}^{n}\left(\alpha_i^* - \alpha_i\right)x_i, \\ \sum_{i=1}^{n}\left(\alpha_i^* - \alpha_i\right) &= 0, \\ C &= \alpha_i^{(*)} + \mu_i^{(*)}. \end{aligned} \tag{7}$$

Substituting Equation (7) into Equation (6) gives the SVR dual problem as:

$$\max_{\alpha,\alpha^*}\sum_{i=1}^{n} y_i\left(\alpha_i^* - \alpha_i\right) - \varepsilon\left(\alpha_i^* + \alpha_i\right) - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\alpha_i^* - \alpha_i\right)\left(\alpha_j^* - \alpha_j\right)K\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right), \tag{8}$$

$$\text{s.t.}\sum_{i=1}^{n}\left(\alpha_i^* - \alpha_i\right) = 0. \tag{9}$$

The above process must satisfy KKT conditions [24], that is:

$$\begin{cases} \alpha_i(\boldsymbol{\omega}\boldsymbol{x}_i + b - y_i - \varepsilon - \xi_i) = 0 \\ \alpha_i^*\left(y_i - \omega\boldsymbol{x}_i - b - \varepsilon - \xi_i^*\right) = 0 \\ \alpha_i\alpha_i^* = 0, \xi_i\xi_i^* = 0 \\ (C - \alpha_i)\xi_i = 0, \left(C - \alpha_i^*\right)\xi_i^* = 0 \end{cases}. \tag{10}$$

Select the component α_j of $\alpha^{(*)}$ in the open interval $(0, C)$ to calculate the parameter b:

$$b = y_j - \sum_{i=1}^{n}\left(\alpha_i^* - \alpha_i\right)K\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) + \varepsilon, \tag{11}$$

Finally, substitute the values of $\boldsymbol{\omega}$ and b into the formula (2):

$$f(\boldsymbol{x}) = \sum_{i=1}^{n}\left(\alpha_i^* - \alpha_i\right)K(\boldsymbol{x}_i, \boldsymbol{x}) + y_j - \sum_{i=1}^{n}\left(\alpha_i^* - \alpha_i\right)K\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) + \varepsilon, \tag{12}$$

which is called the decision function of SVR.

2.2. ACDE Algorithm

Differential evolution (DE) is a stochastic parallel optimization algorithm [25,26] based on population differences proposed by Storn and Price in 1997 [27] on the basis of evolutionary ideas such as the genetic algorithm. DE solves optimization problems by imitating the heuristic swarm

intelligence generated by competition and cooperation among organisms [28]; the algorithm has strong usability, global optimization capability and stability [29] and is widely used in clustering optimization calculations, neural network optimizations, constrained optimization calculations and filter designs. However, the DE algorithm also has two defects: (1) the fixed scaling factor hinders the full search performance of the algorithm when performing mutation operations [30]; (2) in the later stage of optimization, the population diversity is reduced and the algorithm falls easily into local optima [31,32]. Among them, the first defect has been basically solved by Brest et al. 13 years ago. Brest et al. [33]. They assign a separate scaling factor to each individual, so that it can be automatically adjusted according to two thresholds during the running of the algorithm but this will inevitably have some randomness and blindness [34]. To solve these problems, this study uses an improved DE algorithm, ACDE, which improves upon DE in two aspects. First, in the mutation operation stage, the adaptive scaling factor can be changed with the number of iterations, thus addressing the insufficient searching ability caused by the fixed scaling factor. In the standard differential evolution algorithm, increasing the scaling factor can expand the search range and speed up the individual update in the population. The reduction of the scaling factor can enhance the ability of local search and improve the convergence of the algorithm. Based on this, in the early stage of algorithm evolution, the search range can be expanded by increasing the scaling factor. In the later stage of algorithm evolution, the scaling factor is reduced to speed up the convergence of the algorithm. At the same time, in order to avoid the destruction of the genetic structure of the dominant individual, the scaling factor of the individual with higher fitness should be reduced; in order to promote the improvement of the structure of the disadvantaged individual, the scaling factor of the individual with lower fitness should be increased. Second, chaotic motion has the characteristics of sensitivity, randomness, regularity and ergodicity and it can traverse the entire space state without repeating within the specified range. Based on chaos theory, a chaotic fine search strategy is proposed to improve the local search efficiency of the algorithm in order to avoid premature population.

For any optimization problem,

$$\begin{array}{l} \min f(\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N) \\ \text{s.t. } x_j \in \left[L_j, U_j\right], 1 \le j \le N \end{array} \tag{13}$$

where N is the dimension of the solution space and L_j and U_j represent the upper and lower bounds of the range of x_j values, respectively. The specific process of the ACDE algorithm is as follows:

Step 1: Population initialization. D individuals randomly satisfying the constraint condition are generated in the N-dimensional solution space as the i-th individual of the 0-th generation population: $\left\{\boldsymbol{x}_i(0)\middle| x_{j,i}(0) \in \left[L_{j,i}, U_{j,i}\right], i = 1, 2, 3, ..., D; j = 1, 2, 3, ..., N\right\}$. At the same time, the maximum number of iterations is assumed to be M. The value of the j-th dimension of the i-th individual is calculated according to the following formula:

$$x_{j,i}(0) = L_{j,i} + rand(0,1) \cdot \left(L_{j,i} - U_{j,i}\right), \tag{14}$$

where $rand(0,1)$ represents a random number between [0,1] and obeys a uniform distribution.

Step 2: Chaos initialization [13]. Chaotic motion has the characteristics of randomness, regularity and ergodicity and can traverse the entire space state without repetition within a specified range. Generally speaking, the chaotic variable is first mapped to the value interval of the optimization variable and then the chaotic variable is used for optimization. Here, we use the logistic mapping commonly used by chaotic systems, with the basic expression as follows:

$$x_{t+1} = \eta x_t (1 - x_t), \tag{15}$$

where $x_t \in [0,1]$, t is a natural number and $\eta \in [0,4]$; when η= 4, the system is completely chaotic.

Step 3: Mutation. Different from the genetic algorithm, the DE algorithm realizes individual mutation by differential means. That is, two different individuals in the population are selected and the vector difference is scaled to perform vector synthesis with the individuals to be mutated [35]. The k-generation population generated by k iterations is denoted as:

$$\left\{\boldsymbol{x}_i(k)\middle| x_{j,i}(k) \in \left[L_{j,i}, U_{j,i}\right], i = 1,2,3,...,D; j = 1,2,3,...,N\right\}. \tag{16}$$

Three individuals $x_{i1}(k), x_{i2}(k)$ and $x_{i3}(k)$ are randomly selected from the population and the mutation operations are performed according to the following equations:

$$v_i(k+1) = x_{i1}(k) + F(k) \cdot (x_{i2}(k) - x_{i3}(k)), \tag{17}$$

$$F(k) = F_{\min} + (F_{\max} - F_{\min})\frac{F_{i1}(k) - F_{i2}(k)}{F_{i1}(k) - F_{i3}(k)}, \tag{18}$$

where $F(k)$ is the adaptive scaling factor, which controls the influence of the difference vector and changes as the number of iterations k changes; $i1, i2, i3 \in [1, D]$ and $i1 \neq i2 \neq i3 \neq i$; $F_{\min}$ and $F_{\max}$ are the upper and lower limits of the scaling factor, respectively; and $F_{i1}(k)$, $F_{i2}(k)$ and $F_{i3}(k)$ are the optimal, suboptimal and worst-case fitness, respectively, with random selection for the current k-th generation sub-population. Through the mutation operations, an intermediate individual is created:

$$\left\{\boldsymbol{v}_i(k+1)\middle| v_{j,i}(k+1) \in \left[v_{j,i}^L, v_{j,i}^U\right], i = 1,2,3,...,D; j = 1,2,3,...,N\right\}. \tag{19}$$

In the process of evolution, in order to ensure the validity of the solution, it is necessary to judge whether the "genes" in the "chromosome" satisfy the boundary conditions. If the boundary condition is not met, the "gene" is generated in the same way as the initial population, that is, it is regenerated by a random method.

Step 4: Crossover. The inter-individual crossover operation is performed on the g-generation population $\{\boldsymbol{x}_i(k)\}$ and the intermediate individual $\{\boldsymbol{v}_i(k+1)\}$ produced by the mutation operations [36]. The specific method is as follows:

$$\boldsymbol{u}_i(k+1) = \begin{cases} \boldsymbol{v}_i(k+1), \text{ if } rand(0,1) \leq cr \text{ or } j = j_{rand} \\ \boldsymbol{x}_i(k), \quad \text{otherwise} \end{cases}, \tag{20}$$

where cr is the probability of crossover and $cr \in [0,1]$; and j_{rand} is a random integer on the interval [1, 2, 3, ..., N].

Step 5: Selection. Using the principle of greedy choice, the better individuals [37] are selected to form the next generation population according to the following evaluation function:

$$\boldsymbol{x}_i(k+1) = \begin{cases} \boldsymbol{u}_i(k+1), \text{ if } f(\boldsymbol{u}_i(k+1)) \leq f(\boldsymbol{x}_i(k)) \\ \boldsymbol{x}_i(k), \quad \text{otherwise} \end{cases}. \tag{21}$$

Step 6: Chaotic fine search. Letting $\eta = 4$, a chaotic fine search is performed on the decision variable $x_{j,i}^k$ and $x_{j,i}^k$ is mapped to a chaotic variable according to the following formula:

$$s_{j,i}^k = \frac{\left(x_{j,i}^k - L_{j,i}\right)}{\left(U_{j,i} - L_{j,i}\right)}. \tag{22}$$

The next generation iterative chaotic variables $s_{j,i}^{k+1}$ are calculated according to Equation (15) and the chaotic variables $s_{j,i}^{k+1}$ are converted into decision variables $x_{j,i}^{k+1}$ according to Equation (22). Determine whether the chaotic search has reached the maximum number of iterations, if not, continue to iterate.

2.3. ACDE-SVR Prediction Model

One important factor affecting the prediction accuracy of the SVR model is the selection of model parameters. Based on its strong stability, superior global optimization performance and higher convergence speed, this study uses the ACDE algorithm to optimize the parameters of the SVR trajectory prediction model with the penalty factor C, insensitive factor ε and kernel function parameter g. Figure 2 is the flow diagram of ACDE-SVR algorithm. The steps of the ACDE-SVR algorithm are as follows:

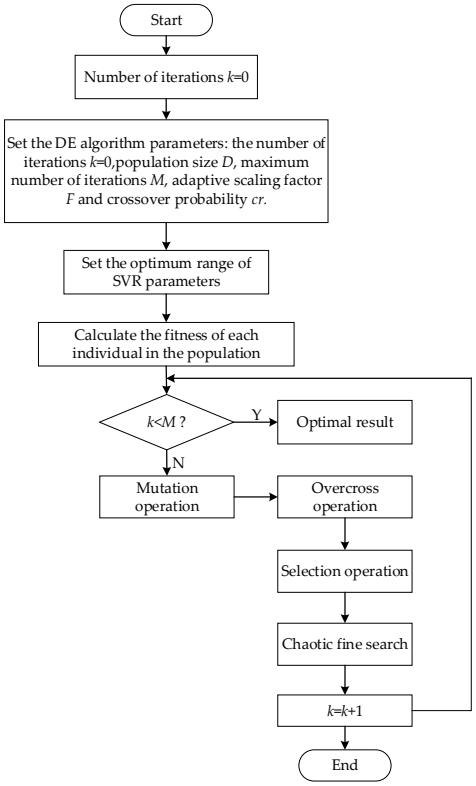


Figure 2. Flow diagram of adaptive chaos differential evolution-support vector regression (ACDE-SVR) algorithm.

Step 1: The trajectory prediction sample is input after de-noising and the current number of iterations is set to k. The ACDE parameters including population size D, maximum number of iterations M, adaptive scaling factor $F(k)$ and crossover probability cr are set according to the actual situation.

Step 2: The optimal range of parameters (C, ε, g) is set and the population is initialized randomly and uniformly.

Step 3: The SVR algorithm is used to predict each individual. The MSE (mean square error) between the predicted value y_i and real value Y_i is calculated according to: $MSE = \frac{1}{l}\sum_{i=1}^{l}(y_i - Y_i)^2$. The MSE is used as the fitness function [38]. The fitness value, individual extremum of each individual $Y_{\text{individual}}(i)$, global extremum Y_{global} and global extremum points X_{global} are recorded.

Step 4: If the number of iterations is less than the maximum number of iterations, that is, $k < M$, the algorithm proceeds to the next step. Otherwise, it proceeds to step 7 and outputs the global extreme point as the searched optimal parameter, $(C^*, \varepsilon^*, g^*)$.

Step 5: According to Equations (17)–(22), mutation, crossover, selection and chaotic fine search operations are carried out to generate new populations. The fitness of individuals within the population, individual extremum $Y_{\text{individual}}(i)$, global extremum Y_{global} and global extremum X_{global} are calculated.

Step 6: Iterations are set as $k = k + 1$ and the algorithm returns to step 4.

Step 7: The SVR prediction model is built based on the calculated parameters.

2.4. Setting of Parameters in ACDE Algorithm

For the ACDE algorithm, the parameters should be reasonably set to obtain better parameter optimization results. The main parameters of the ACDE algorithm are scaling factor F, population size D, maximum iteration number M and crossover probability cr. Among them, the selection of scaling factor is introduced in Part 2.2, which is related to the number of iterations k. The bases for setting the remaining three parameters are as follows:

(1) Population size D. According to experience, D should be 5 to 10 times the dimension of the problem but not less than 4; otherwise, the mutation operation will not be possible. Generally speaking, as D increases, the population diversity will gradually increase along with the probability of reaching an optimal solution. However, this will also greatly increase the calculation time. In general, D ranges from 20 to 50.

(2) Maximum number of iterations M. This parameter is the termination condition of the evolutionary operation. A larger M increases running time of the program and yields a more accurate optimal solution. However, when M is too large, the accuracy of the optimal solution will not increase correspondingly, because the population has reached an extremely single state at this time and the algorithm cannot jump out of the local optimal solution [39].

(3) Crossover probability cr. The value range of this parameter is [0, 1] and the value range of this parameter is (0.0, 0.2) when the objective function is separable [40]. If cr is large, the population convergence is gradually accelerated and precocity is prone to occur.

2.5. ACDE Algorithm Superiority Verification

In order to verify the excellent global optimization performance of the proposed ACDE algorithm, which is compared with the current state-of-the-art differential evolution algorithm, the LSHADE algorithm [41]. The ACDE algorithm and LSHADE algorithm are evaluated using a set of problems presented in the CEC2017 competition on single objective bound constrained real-parameter optimization. This benchmark contains 30 test functions with multiple characteristics. T is the dimensionality of the problem and the functions are tested on 50T. In short, functions 1–3 are unimodal, functions 4–10 are multimodal, functions 11–20 are hybrid functions and 21–30 are composition functions. More details can be found in Reference [42].

The parameters values of ACDE algorithm are set as follows: population size D of the ACDE algorithm set to 50, maximum iteration number M set to 10,000*T, the upper and lower limits of the scaling factor are 1.2 and 0.3 and crossover probability cr set to 0.9. While the parameters of the LAHADE algorithm are set as shown in Reference [41].

The two algorithms perform 51 simulation tests on 30 benchmark problems, respectively. The statistical results of the comparisons on the benchmarks with ACDE algorithm and LSHADE algorithm are summarized in Table 1. It includes the mean and the standard deviations of error. The best results are marked in bold for all problems. As can be seen from Table 1, the ACDE algorithm is significantly better or flatter than the LSHADE algorithm on 18 functions. The ACDE algorithm is less accurate than the LSHADE algorithm on 12 functions. In general, the optimization performance of the ACDE algorithm is slightly higher than the LSHADE algorithm.

Table 1. Performance comparison of ACDE and LSHADE algorithms of the 50T benchmark.

	Function	ACDE		LSHADE	
		Mean	**Std.**	**Mean**	**Std.**
Unimodal Functions	F1	**0.00**	**0.00**	**0.00**	**0.00**
	F2	**0.00**	**0.00**	**0.00**	**0.00**
	F3	**0.00**	**0.00**	**0.00**	**0.00**
Simple Multimodal Functions	F4	5.14×10^{1}	4.43×10^{1}	$\mathbf{5.64 \times 10^{-1}}$	$\mathbf{1.46 \times 10^{0}}$
	F5	$\mathbf{2.52 \times 10^{1}}$	$\mathbf{1.41 \times 10^{0}}$	6.19×10^{0}	2.08×10^{0}
	F6	8.15×10^{-8}	1.09×10^{-6}	**0.00**	**0.00**
	F7	$\mathbf{4.12 \times 10^{1}}$	$\mathbf{1.00 \times 10^{0}}$	5.67×10^{1}	1.15×10^{0}
	F8	$\mathbf{2.63 \times 10^{1}}$	$\mathbf{1.58 \times 10^{0}}$	5.95×10^{0}	2.24×10^{0}
	F9	**0.00**	**0.00**	**0.00**	**0.00**
	F10	3.10×10^{2}	3.41×10^{1}	$\mathbf{3.62 \times 10^{0}}$	$\mathbf{6.71 \times 10^{0}}$
	F11	$\mathbf{2.14 \times 10^{0}}$	$\mathbf{1.08 \times 10^{0}}$	9.07×10^{0}	3.36×10^{0}
	F12	$\mathbf{1.48 \times 10^{0}}$	$\mathbf{3.65 \times 10^{0}}$	1.58×10^{0}	4.42×10^{0}
	F13	6.94×10^{1}	3.45×10^{1}	$\mathbf{3.43 \times 10^{0}}$	$\mathbf{1.50 \times 10^{0}}$
	F14	$\mathbf{2.41 \times 10^{1}}$	$\mathbf{2.47 \times 10^{0}}$	2.90×10^{1}	3.94×10^{0}
	F15	2.56×10^{1}	4.06×10^{0}	$\mathbf{1.56 \times 10^{1}}$	$\mathbf{6.70 \times 10^{0}}$
	F16	$\mathbf{2.76 \times 10^{2}}$	$\mathbf{9.73 \times 10^{1}}$	4.13×10^{2}	1.68×10^{2}
Hybrid Function	F17	$\mathbf{2.09 \times 10^{2}}$	$\mathbf{7.32 \times 10^{1}}$	2.94×10^{2}	1.20×10^{2}
	F18	$\mathbf{2.58 \times 10^{1}}$	$\mathbf{2.55 \times 10^{0}}$	3.30×10^{1}	7.23×10^{0}
	F19	$\mathbf{1.74 \times 10^{1}}$	$\mathbf{2.48 \times 10^{0}}$	2.18×10^{1}	4.82×10^{0}
	F20	$\mathbf{1.14 \times 10^{2}}$	$\mathbf{3.95 \times 10^{1}}$	1.65×10^{2}	1.11×10^{2}
	F21	2.67×10^{2}	8.42×10^{0}	$\mathbf{2.15 \times 10^{2}}$	$\mathbf{9.23 \times 10^{0}}$
	F22	2.13×10^{3}	2.17×10^{3}	$\mathbf{8.17 \times 10^{2}}$	$\mathbf{1.49 \times 10^{3}}$
Composition Function	F23	$\mathbf{4.39 \times 10^{2}}$	$\mathbf{6.95 \times 10^{0}}$	4.39×10^{2}	7.19×10^{0}
	F24	5.13×10^{2}	5.67×10^{0}	$\mathbf{5.12 \times 10^{2}}$	$\mathbf{5.80 \times 10^{0}}$
	F25	4.97×10^{2}	2.45×10^{0}	$\mathbf{4.63 \times 10^{2}}$	$\mathbf{8.50 \times 10^{0}}$
	F26	$\mathbf{1.07 \times 10^{3}}$	$\mathbf{5.43 \times 10^{1}}$	1.14×10^{3}	$6.93\text{E} \times 10^{1}$
	F27	$\mathbf{4.18 \times 10^{2}}$	$\mathbf{9.94 \times 10^{0}}$	5.03×10^{2}	8.99×10^{-5}
	F28	4.59×10^{2}	1.28×10^{1}	$\mathbf{4.04 \times 10^{5}}$	$\mathbf{8.49 \times 10^{0}}$
	F29	$\mathbf{1.24 \times 10^{2}}$	$\mathbf{9.14 \times 10^{0}}$	2.75×10^{2}	5.96×10^{1}
	F30	6.55×10^{3}	5.15×10^{2}	$\mathbf{6.57 \times 10^{2}}$	$\mathbf{4.10 \times 10^{2}}$

3. AIS Sensor Data Acquisition and Preprocessing

3.1. AIS Sensor Navigation Trajectory Data Extraction and Separation

In contrast to actual maritime navigation, the Automatic Identification System (AIS) trajectory data of a target ship can be directly obtained as sample data. In the simulation experimental stage, the route data from within the massive AIS sensor data from a certain sea area must be extracted to screen out ship trajectories that meets the model requirements. The shore-based AIS has navigational data for all ships passing through the sea and a ship may travel back and forth several times between the starting and end points of a certain area. Thus, the route data are numerous and complicated and it is difficult to directly extract ship routes to create a sample set for model training.

The trajectories of different ships are distinguished according to their unique maritime mobile service identification (MMSI) numbers. According to reference [43], the time intervals between two adjacent data points of the same ship were preliminarily calculated and the route trajectories were identified by the speed and time interval. Data points with long time intervals and instantaneous velocities of 0 or close to 0 were selected as the starting points of the route. Considering the abnormal recording of time data, the starting point of the route was determined by combining the speed and change in ship position based on a time difference judgment. An AIS sequence is denoted as $T = \{(t_1, v_1, \lambda_1, ...), (t_2, v_2, \lambda_2, ...), ..., (t_n, v_n, \lambda_n, ...)\}$, where t_n denotes the time interval from the next

point, v_n denotes the speed of ship and λ_n denotes the longitude of the ship. The specific algorithm for extracting trajectory data is as follows:

Step 1: In sequence T, t_i is traversed until data points t_i and t_j satisfy $\left|t_i - t_j\right| \geq 12h$. Then, the i and j data points in the time series are set as sequence tangent points and the steps are repeated until all data points in the sequence are traversed.

Step 2: If there is an abnormality in the time data, this step is used for assistance. In sequence T, v_i is traversed until the data point satisfies $v_i \approx 0$; this point is then set to the tangent point in the sequence. Step 1 is then repeated until all data points in the sequence have been traversed.

3.2. AIS Sensor Data Cleaning

AIS sensor data unavoidably contains data anomalies, missing data and other issues that necessitate cleaning of the data after trajectory extraction. After trajectory extraction, the original cluttered AIS data are processed into trajectory data sets. In each trajectory data set, if the acquisition time of two data points is quite different, this indicates that data in the sequence are missing and must be interpolated. According to the experimental error results of linear interpolation, Lagrange interpolation, median interpolation and mean interpolation discussed in reference [42], the Lagrange interpolation method has the lowest interpolation error for missing data. Therefore, the Lagrange interpolation method was used to interpolate missing values of ship latitude, longitude, heading and speed.

3.3. Wavelet Threshold De-Noising

The AIS system obtains ship trajectory information by accessing GPS signals. Owing to the mutual interference and signal attenuation of maritime communication equipment, ship position information may also be affected by noise interference and data distortion.

For a long time, people often choose Fourier transform to process signals. However, because the function constructed by Fourier transform is periodic sine wave and cosine wave, it can only be applied to those signals with periodic or approximate periodicity but the effect is not very good on those signals with non-periodic or strong local characteristics.

The wavelet transform developed from Fourier transform can solve the above problems well. It not only retains many advantages of Fourier transform but also improves and develops on the original basis, so that it can process signals in time-frequency domain. The remarkable advantage of wavelet transform is that it can process the signal more subtly and show some characteristics of the signal better. It realizes the requirement of localization and multi-scale analysis of the signal in time-frequency domain. The signal de-noising method developed on the basis of wavelet shows good de-noising effect, which is the perfection and development of Fourier transform in the field of signal processing. Based on this, we choose the wavelet threshold de-noising method based on wavelet analysis theory. The advantage of this method is that the noise is almost completely suppressed and the characteristic peaks of the original signal are well preserved. Moreover, de-noising using soft threshold can minimize the maximum mean square error of the estimated signal, that is, the estimated signal after de-noising is the approximate optimal estimate of the original signal and the estimated signal is at least as smooth as the original signal without additional oscillation. In addition, the method is fast in calculation and has wide adaptability and is the most widely used one of many wavelet de-noising methods.

The main theoretical basis of wavelet threshold de-noising is as follows. The wavelet transform has a strong decorrelation, which concentrates the energy of the signal in some small wavelet coefficients in the wavelet domain, while the energy of the noise is distributed in the entire wavelet domain. Therefore, after the wavelet decomposition, the amplitude of the wavelet coefficient of the signal is greater than the amplitude of the coefficient of the noise. Wavelet coefficients with larger amplitudes can be considered to be generally dominated by signals, while coefficients with smaller amplitudes are largely noise. Thus, the threshold can be used to preserve the signal coefficients, while reducing the noise to almost zero.

3.4. Data Normalization

To accelerate model convergence and solve errors caused by large differences in magnitude between model samples, the sample data were linearly normalized by the following formula to the interval [0, 1]:

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}, \tag{23}$$

where x is the sample data and $x \in \mathbf{R}^n$; x' is the normalized data and $x' \in [0, 1]$; and $x_{\max}$ and $x_{\min}$ are the maximum and minimum values in the sample points, respectively.

4. Trajectory Prediction Model Based on AIS Data and ACDE-SVM

4.1. Trajectory Prediction Model

When a ship is sailing, it mainly determines its navigational behavior by receiving the AIS data of a target ship to make timely and accurate collision avoidance decisions under harsh sea conditions and complicated encounter situations. In an actual voyage, the navigational behavior of different ships is mainly reflected in the time series changes in ship position, heading and speed. The navigational behavior of a ship at time t can be characterized as

$$N_t = \{lon_t, lat_t, C_t, v_t, T_t\}, \tag{24}$$

where lon_t, lat_t, C_t, v_t and T_t represent the longitude, latitude, course, speed over ground and Unix time of the ship, respectively, at time t. Aiming at the SVM characteristics of multivariable input and single variable output, the ship longitude and latitude parameters should be separately optimized. That is to say, in order to separately predict the longitude and latitude of the ship characterizing the nature of the ship, two SVR prediction models need to be established. The input characteristics of the two models are the same and the output characteristics are different. Generally speaking, the future navigational behavior of a ship is often the result of the interaction between current and previous navigational behaviors. Therefore, in order to improve the accuracy of trajectory prediction, the navigation behavioral of the past four moments, $N_{t-3}, N_{t-2}, N_{t-1}$ and N_t and the time stamp of the next moment, $t + 1$, were used as input variables and the longitude and latitude of the next moment were used as the model output variables respectively:

$$\begin{aligned} Input &= \{N_{t-3}, N_{t-2}, N_{t-2}, N_t, T_{t+1}\} \\ Output_1 &= \{lon_{t+1}\} \\ Output_2 &= \{lat_{t+1}\} \end{aligned} \tag{25}$$

where $N_{t-3}, N_{t-2}, N_{t-1}$ and N_t represent the navigational behaviors at times $t - 3$, $t - 2$, $t - 1$ and t, respectively, from Equation (24).

The model uses the root mean square error (*RMSE*), mean absolute error (*MAE*) and maximum absolute error (E_{MAX}) to measure the prediction accuracy, which are calculated as:

$$RMSE = \sqrt{\frac{1}{S}\sum_{t=1}^{S}(y_t - Y_t)^2}, \tag{26}$$

$$MAE = \frac{1}{S}\sum_{i=1}^{N}\left|y_t - Y_t\right|, \tag{27}$$

$$E_{MAX} = \max\left|y_t - Y_t\right|, \tag{28}$$

where S represents the total sample size and y_t and Y_t represent the predicted and actual values, respectively.

4.2. Analysis of Experimental Process

This study used the libsvm toolbox developed by Professor Lin Chih-Jen [44] of Taiwan University for modeling and simulations. AIS trajectories were extracted from 2,346,643 sets of AIS data from Tianjin Port waters in March 2015 using the algorithm described in Section 2.1. A certain trajectory of the ship with an MMSI of 538,004,758 was randomly selected as the research object; the trajectory was cleaned and missing values were interpolated. To satisfy the collision avoidance requirements, it was found that there may have been a collision risk with the target ship during maritime navigation. The AIS data of the target ship acquired in real time were used as the sample data to establish a prediction model, for which there may be a problem related to the small number of training samples. To simulate accurately, a part of the trajectory data was selected as the sample and 226 sets of AIS data were selected as the sample data. Figure 3 is ship trajectory with sample data sets. Among these, 200 groups were used as the training set and 26 groups were used as the test set to carry out simulation experiments. The sample data structure is summarized in Table 2.

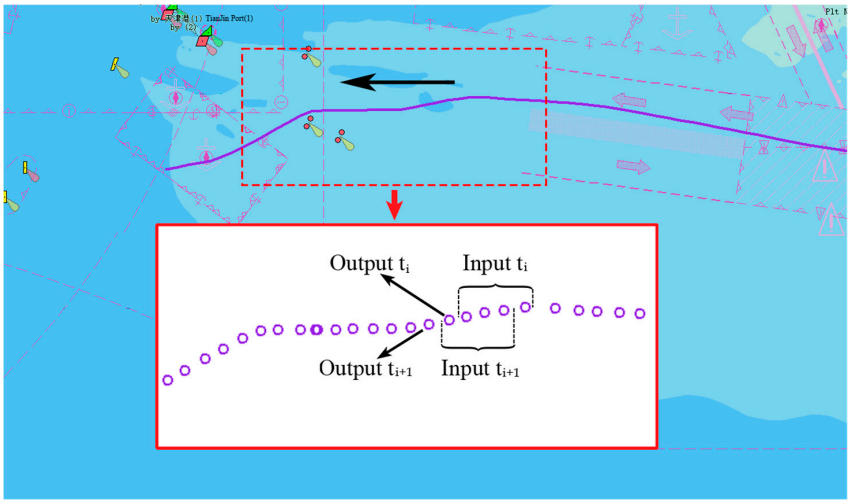


Figure 3. Ship trajectory with sample data sets. The purple line refers to the Automatic Identification System (AIS) trajectory of ship. The purple hollow circle refers to the trajectory point of the selected trajectory segment and the arrow indicates the ship's navigation direction. Input t_i refers to the trajectory point of the current t_i time and the past three times (t_{i-1}, t_{i-2}, t_{i-3}) as the input data of the model and output t_i refers to the trajectory of the next time t_{i+1} The point is used as the output data of the model; Input t_{i+1} refers to the track point of the t_{i+1} time and the past three times (t_i, t_{i-1}, t_{i-2}) as the input data of the model and the output t_{i+1} means The track point of t_{i+2} at the next moment is used as the output data of the model.

Table 2. Sample data structure.

Serial Number	Ship Name	MMSI	Course/(°)	Speed/(kn)	Longitude/(°)	Latitude/(°)	Unix Time/(s)
1	STELLA JADE	538004758	279.1	11.9	120.019533	38.710583	1,426,284,921
2	STELLA JADE	538004758	278.2	11.8	119.934217	38.720833	1,426,286,160
3	STELLA JADE	538004758	278.5	11.8	119.886767	38.726333	1,426,286,873
4	STELLA JADE	538004758	278.4	11.7	119.86145	38.729283	1,426,287,221
5	STELLA JADE	538004758	278.3	11.7	119.838133	38.731967	1,426,287,560
6	STELLA JADE	538004758	278.4	11.7	119.8326	38.732633	1,426,287,641
7	STELLA JADE	538004758	278.5	11.7	119.813617	38.734850	1,426,287,920

The original sequence was constructed from 226 sets of ship position data from AIS information and de-noised by wavelet threshold. Owing to the strong continuity of the ship sequence, the sym8 wavelet basis function, with strong continuity and symmetry, was used for threshold de-noising. The number of noise reduction times was set to 3, the heuristic threshold function was used to select the threshold and the soft threshold function was used for wavelet de-noising. Figure 4 shows a noise sequence and Figure 5 is a comparison between original and de-noised sequences. As can be seen from Figure 5, the presence of noise causes a slight disturbance in the position signal. To verify the effectiveness of wavelet de-noising, we used ship position sequences with and without de-noising to predict ship trajectories and compared the results. The comparison shows that the data predicted with de-noising are more accurate.

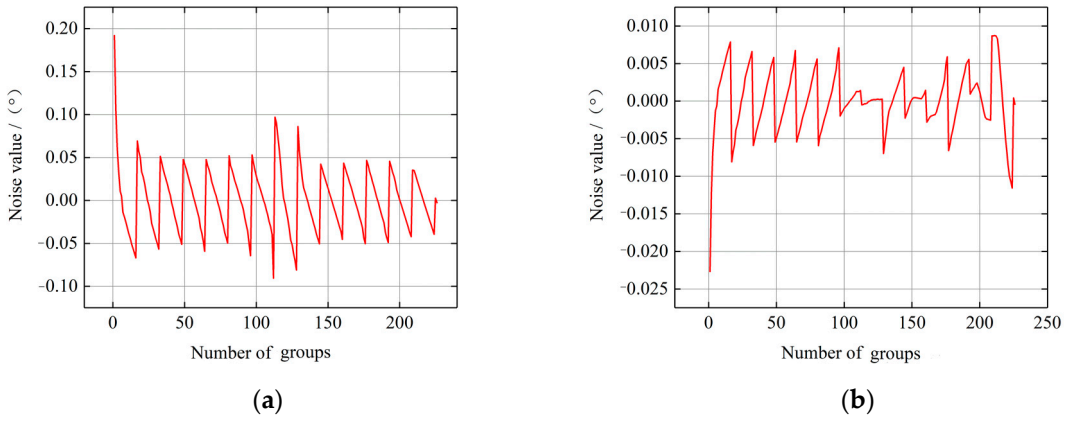


Figure 4. Example noise sequences. (**a**) Longitude noise sequence; (**b**) Latitude noise sequence.

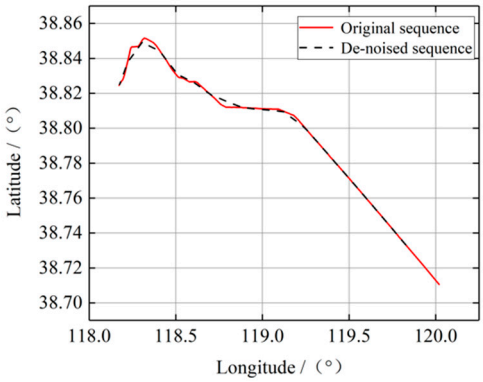


Figure 5. Comparison of original and de-noised sequences.

To reduce the influence of the order of magnitude on prediction accuracy, the training set and test set data were normalized to the interval [0, 1] and the RBF was selected as the kernel function. The ACDE algorithm was used to optimize the parameters of the longitude and latitude prediction models and the optimal penalty factor C, insensitive factor ε and kernel function parameter g were searched. According to Section 2.4, the parameters of the ACDE algorithm were set as follows: population size D of the ACDE algorithm set to 30, maximum iteration number M set to 200, the upper and lower limits of the scaling factor are 1.2 and 0.3 and crossover probability cr set to 0.9. The optimization results are summarized in Table 3. As can be seen from Table 3, the parameters of the two models are different. As shown in Equation (24), the input variables of the two models are the same but the output variables are different and the difference of the output variables determines the difference between the optimized parameters of the two models.

Table 3. Parameter optimization results of the ACDE algorithm.

Parameter	Penalty Factor C	Insensitive Factor ε	Kernel Function Parameter g
Latitude	20.7043	0.016	652.1897
Longitude	81.4909	0.0075	905.7929

4.3. Results Analysis

4.3.1. Verification of De-Noising Effect

To verify the de-noising effect, trajectory sequences with and without de-noising were used for trajectory prediction according to the above prediction steps and the errors were compared. The results are summarized in Table 4.

Table 4. Comparison of trajectory sequence prediction errors with and without de-noising.

Sample Data		Maximum Absolute Error E_{MAX}/(°)	Mean Absolute Error MAE/(°)	Mean Square Error $RMSE$/(°)
With de-noising	Longitude	3.475414×10^{-5}	1.216385×10^{-5}	1.562460×10^{-5}
	Latitude	3.957327×10^{-5}	1.345339×10^{-5}	1.660051×10^{-5}
Without de-noising	Longitude	6.785011×10^{-5}	4.477475×10^{-5}	2.201641×10^{-5}
	Latitude	7.387650×10^{-5}	5.524124×10^{-5}	2.932073×10^{-5}

4.3.2. Comparison of Prediction Results from Different Optimization Algorithms

To verify the superiority of the ACDE algorithm for optimization of SVR trajectory prediction model parameters, parameters determined by other optimization methods were used for comparison. Parameters optimized by the standard DE algorithm, PSO (particle swarm optimization) algorithm, grid search algorithm and GA (genetic algorithm) and the default parameters of the libsvm toolbox were used for trajectory prediction simulations. The predicted results were compared and analyzed and the error indicator values are given in Table 5. First, the prediction results of the ACDE-SVR model are compared with the standard DE-SVR model. The ACDE-SVR algorithm converges faster than the DE-SVR algorithm and each error index is smaller than that of the DE-SVR, which indicates that the ACDE-SVR trajectory prediction model has the advantages of faster convergence and higher prediction accuracy. In addition, the ship trajectory prediction models based on the parameters optimized by grid search and default toolbox values require less run time but have low prediction accuracies. Using parameters optimized by GA and PSO algorithms to build the model greatly improves the prediction accuracy but also increases the convergence time. The model based on the parameters optimized by the ACDE algorithm has the highest prediction accuracy, lowest run time and the best overall performance.

Table 5. Influence of different parameter optimization methods on prediction accuracy of SVR model.

Parameter Optimization Method for SVR Model		E_{MAX}/(°)	MAE/(°)	$RMSE$/(°)	Run Time/s
ACDE	Longitude	5.475414×10^{-5}	3.216385×10^{-5}	3.562460×10^{-5}	38.14
	Latitude	5.957327×10^{-5}	3.345339×10^{-5}	3.660051×10^{-5}	
DE	Longitude	6.749311×10^{-5}	4.314214×10^{-5}	4.465213×10^{-5}	45.21
	Latitude	6.964512×10^{-5}	4.657491×10^{-5}	5.041521×10^{-5}	
PSO	Longitude	7.495242×10^{-5}	5.745210×10^{-5}	5.954213×10^{-5}	74.59
	Latitude	6.935740×10^{-5}	5.526796×10^{-5}	5.631453×10^{-5}	
Grid search	Longitude	1.465548×10^{-4}	1.642193×10^{-4}	2.146923×10^{-4}	24.71
	Latitude	1.941526×10^{-4}	2.145237×10^{-4}	3.461430×10^{-4}	
GA	Longitude	7.041523×10^{-5}	5.042750×10^{-5}	5.546311×10^{-5}	87.91
	Latitude	7.352207×10^{-5}	6.004726×10^{-5}	6.421012×10^{-5}	
Default toolbox values	Longitude	1.942751×10^{-4}	1.945221×10^{-4}	3.047522×10^{-4}	6.77
	Latitude	2.545720×10^{-4}	2.541201×10^{-4}	3.948753×10^{-4}	

4.3.3. Comparison of Prediction Results between ACDE-SVR Model and Neural Network Prediction Model

Sections 4.3.1 and 4.3.2 respectively verify that wavelet threshold de-noising and ACDE algorithm optimization yield an SVR model with a higher prediction accuracy; thus, the model was used to predict the trajectory sample data. A comparison between predicted results and actual results is shown in Figure 6 and the longitude and latitude prediction errors are shown in Figure 7, where the error value refers to the absolute error, that is, the difference between the predicted value and actual value. It can be seen from Figures 6 and 7 that the results of trajectory prediction using the ACDE-SVR algorithm are basically consistent with the actual values. The magnitude of the longitude and latitude prediction error is 10^{-5} and the maximum absolute values of 20 points longitude and latitude, respectively, are 3.475414×10^{-5}° (3.86 m) and 3.957327×10^{-5}° (4.394 m). This error is within the acceptable range and thus the prediction accuracy satisfies the requirements for making collision avoidance decisions during actual maritime navigation.

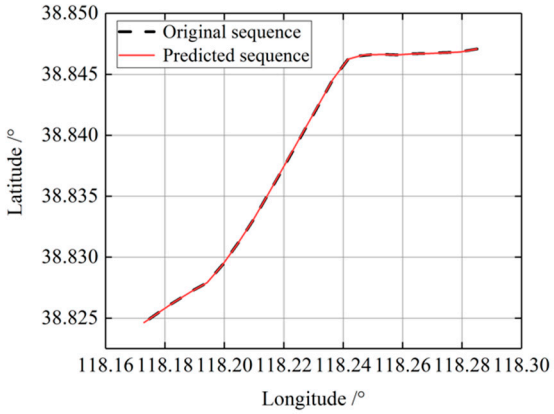


Figure 6. Comparison between predicted and original data.

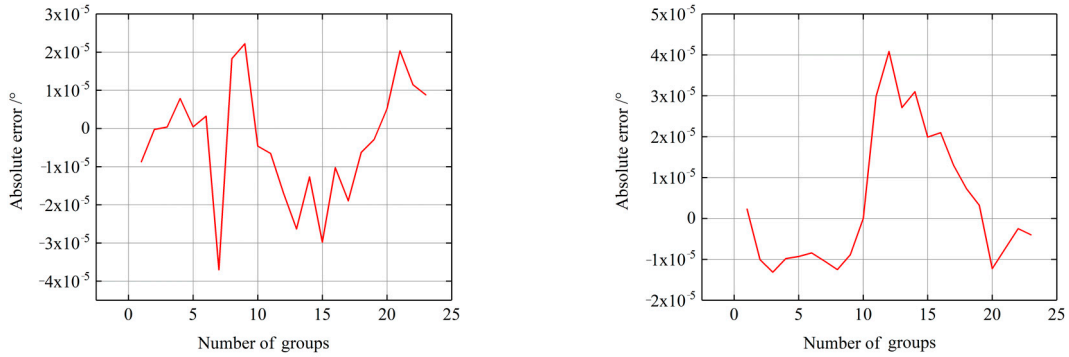


Figure 7. Error values of trajectory prediction model based on ACDE-SVR. (**a**) Longitudinal error values; (**b**) Latitudinal error values.

To verify the prediction accuracy of the model, the BP neural network, recurrent neural network (RNN) and SVR prediction models were also compared. As the preferred neural network for processing time series data, RNN has good timing prediction performance [45–47]. Because the initial weights and thresholds of each BP neural network layer are set randomly [12], it has some problems such as a low prediction accuracy and ease of falling into local optima. To solve the influence of initial parameter settings on the BP neural network prediction model, the ACDE optimization algorithm was used to optimize the weights and thresholds of the BP neural network. In summary, the RNN trajectory prediction model and ACDE-BP neural network model were established under the premise

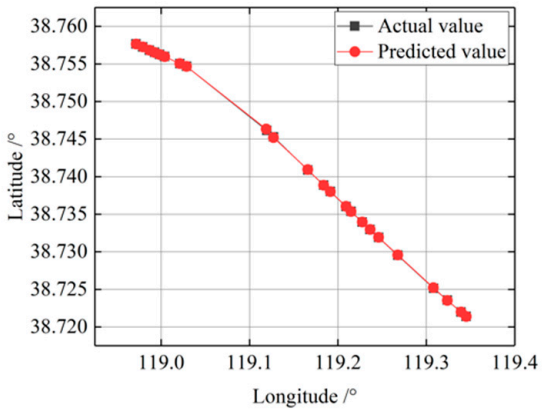
of limited data samples and the prediction errors were calculated and compared with the ACDE-SVR prediction results as shown in Table 6. Under the premise of limited sample data, the ACDE-SVR trajectory prediction model has the highest training accuracy and the shortest run time. Therefore, the ACDE-SVR trajectory prediction model proposed in this paper has good prediction results in regard to both prediction accuracy and operation time.

Table 6. Comparison of prediction error indicators between three trajectory prediction models.

Model		E_{MAX}/(°)	*MAE*/(°)	*RMSE*/(°)	Time/s
ACDE-SVR	Longitude	5.475414×10^{-5}	3.216385×10^{-5}	3.562460×10^{-5}	33.51
	Latitude	5.957327×10^{-5}	3.345339×10^{-5}	3.660051×10^{-5}	
ACDE-BP	Longitude	7.484621×10^{-5}	5.257410×10^{-5}	5.745215×10^{-8}	85.94
	Latitude	8.145272×10^{-5}	5.547871×10^{-5}	5.945120×10^{-8}	
RNN	Longitude	7.124120×10^{-5}	4.945211×10^{-5}	5.457329×10^{-5}	65.41
	Latitude	7.844572×10^{-5}	5.247560×10^{-5}	5.842124×10^{-5}	

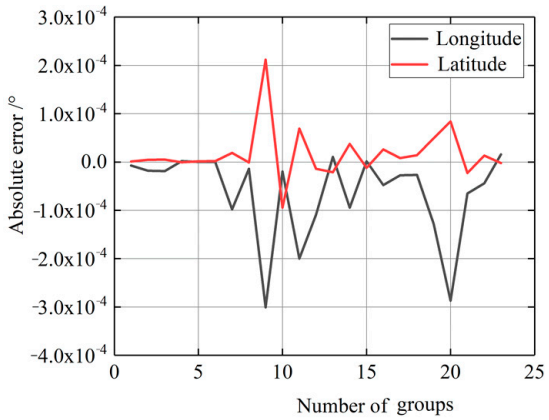
4.3.4. Comparison of Different Trajectory Sequences

To verify the generalization performance of the model, the ACDE-SVR algorithm was used to predict and compare different trajectories. Six trajectory sequences from different ships were randomly selected and some of the six trajectories were arbitrarily selected as sample data for model training and prediction. In order to facilitate the comparison of the prediction results of different trajectories, the six ship trajectories are respectively marked as a-f trajectories. Comparisons between the predicted values of the six trajectories and actual values are shown in Figures 8–13 and the specific prediction errors are shown in Table 7. It can be roughly seen from Figures 8–13 that the prediction error orders of magnitude are −4 or −5 and the predicted values are essentially consistent with the actual values. To further analyze the prediction errors of the different trajectory sequences, the analysis in Table 6 was carried out, which clearly shows that the prediction error of the a-trajectory is the largest. The MAE of latitude and longitude is 3.007×10^{-4}° (33.41 m), which is within the allowable range and can thus meet the collision avoidance requirement. The other trajectory errors are smaller than that of the a-trajectory and thus can also meet the collision avoidance requirement. In summary, the trajectory prediction model based on ACDE-SVR has high generalization capability and can be applied for collision avoidance.



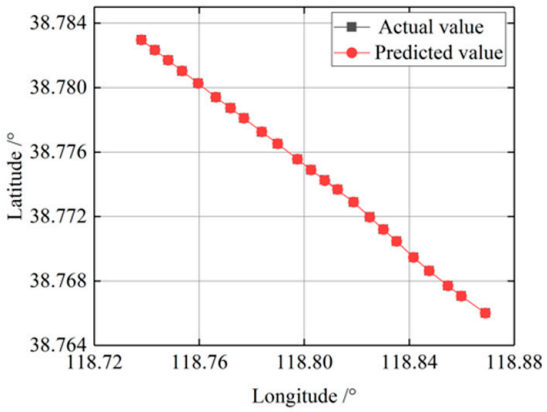
(**a**)

Figure 8. *Cont.*

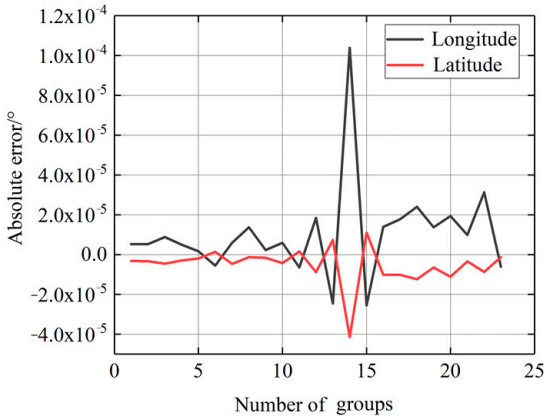


(**b**)

Figure 8. Prediction results of a-trajectory using the ACDE-SVR model. (**a**) Comparison of predicted and actual values of a-trajectory; (**b**) Longitude and latitude absolute error values of a-trajectory.

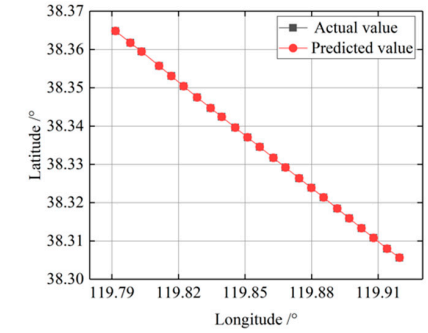


(**a**)

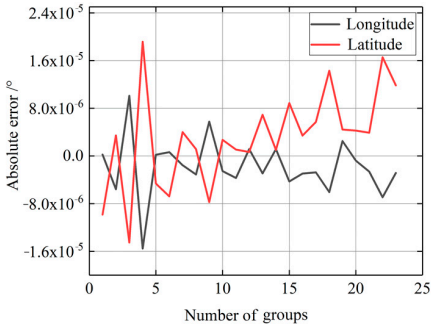


(**b**)

Figure 9. Prediction results of b-trajectory using the ACDE-SVR model. (**a**) Comparison of predicted and actual values of b-trajectory; (**b**) Longitude and latitude absolute error values of b-trajectory.

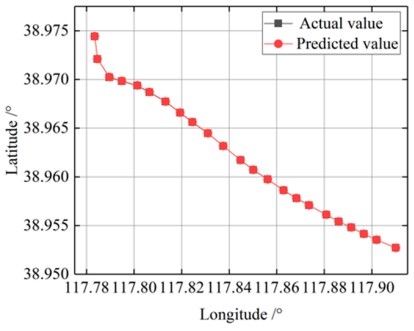


(**a**)

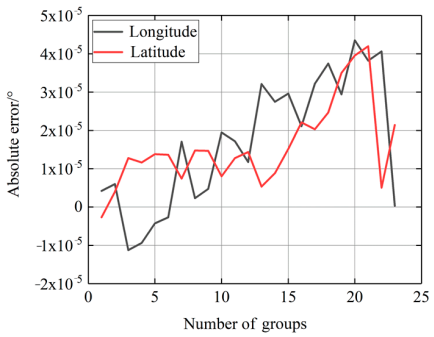


(**b**)

Figure 10. Prediction results of c-trajectory using the ACDE-SVR model. (**a**) Comparison of predicted and actual values of c-trajectory; (**b**) Longitude and latitude absolute error values of c-trajectory.

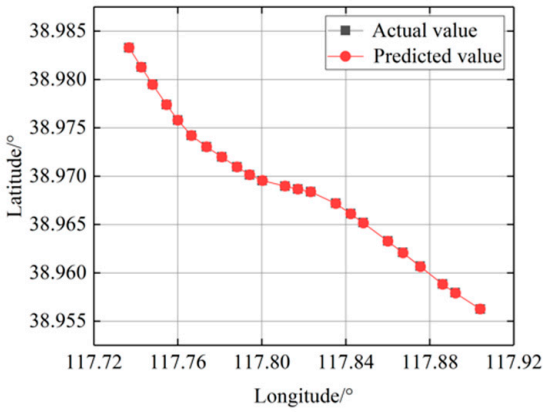


(**a**)

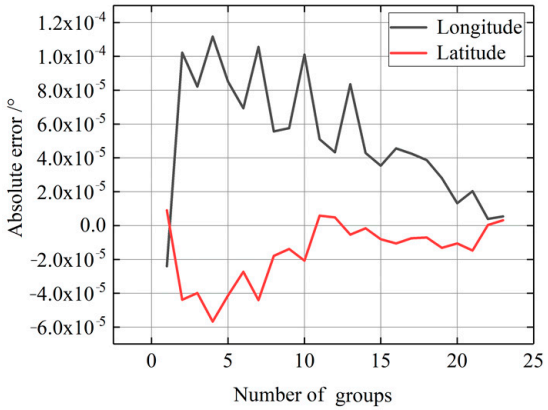


(**b**)

Figure 11. Prediction results of d-trajectory using the ACDE-SVR model. (**a**) Comparison of predicted and actual values of d-trajectory; (**b**) Longitude and latitude absolute error values of d-trajectory.

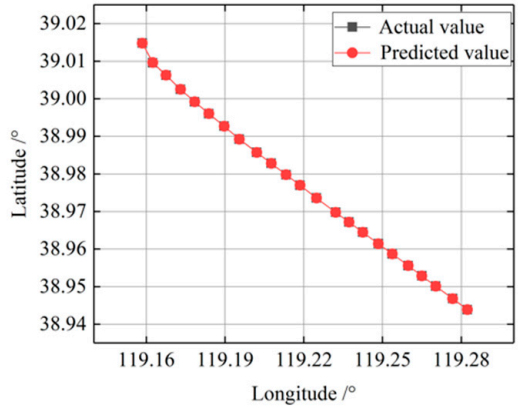


(**a**)



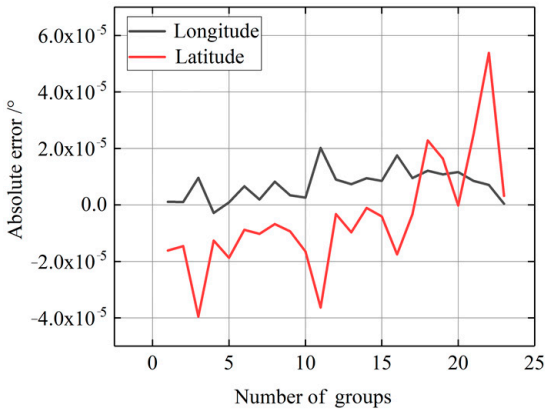
(**b**)

Figure 12. Prediction results of e-trajectory using the ACDE-SVR model. (**a**) Comparison of predicted and actual values of e-trajectory; (**b**) Longitude and latitude absolute error values of e-trajectory.



(**a**)

Figure 13. *Cont*.



(**b**)

Figure 13. Prediction results of f-trajectory using the ACDE-SVR model. (**a**) Comparison of predicted and actual values of f-trajectory; (**b**) Longitude and latitude absolute error values of f-trajectory.

Table 7. Comparison of prediction results for different ship trajectory sequences.

Serial Letter	Ship Name	Training Set/Group	Test Set/Group	E_{MAX}/(°)		MAE/(°)		Time/s
				Longitude	Latitude	Longitude	Latitude	
a	ANASA	143	26	3.007×10^{-4}	2.119×10^{-4}	3.470×10^{-5}	5.048×10^{-5}	46.5
b	BULK INDIA	159	26	1.025×10^{-4}	4.090×10^{-5}	1.593×10^{-5}	6.592×10^{-6}	50.7
c	KAI ZHOU1	184	26	2.288×10^{-5}	1.655×10^{-5}	3.736×10^{-6}	6.817×10^{-6}	62.1
d	CLYDE	171	26	4.317×10^{-5}	3.581×10^{-5}	1.846×10^{-5}	1.186×10^{-5}	60.8
e	DE PU	182	26	1.117×10^{-4}	5.676×10^{-5}	1.575×10^{-5}	4.347×10^{-5}	64.4
f	Star sirius	123	26	2.019×10^{-5}	3.953×10^{-5}	7.394×10^{-6}	1.522×10^{-5}	41.3

5. Conclusions

This paper analyzes and summarizes the problems existing in current ship trajectory prediction methods. Considering the good global optimal fitting performance of SVR and its characteristics suitable for small sample training, an offline trajectory prediction method based on SVR was proposed. The speed, course, longitude, latitude and Unix time from AIS information were selected as the sample characteristic variables in the model to predict the time series of a ship's trajectory toward a better understanding of current and future trends of other ships and generating collision avoidance decisions with a certain foresight. However, the model proposed in this study is an offline model, which assumes that all samples are acquired at one time and cannot be modified once the model has been trained. AIS information is collected intermittently based on speed and heading. If newly acquired AIS data are significantly different from the original sample data, the prediction error is large. Therefore, establishing an incremental SVR-based trajectory prediction model is the next step that should be explored.

Author Contributions: Conceptualization, J.L. and K.Z.; methodology, J.L.; software, J.L. and K.Z.; validation, J.L., G.S. and K.Z.; formal analysis, J.L.; investigation, J.L.; resources, J.L.; data curation, J.L. and K.Z.; writing—original draft preparation, J.L.; writing—review and editing, J.L., G.S. and K.Z.; visualization, X.X.; supervision, G.S. and K.Z.

Funding: This research was funded by the National Natural Science Foundation of China, grant number 51579025; the Natural Science Foundation of Liaoning Province, grant number 20170540090; and the Fundamental Research Funds for the Central Universities, grant number 3132018306.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ren, P. A Study on Ship Collision Avoidance Decision-Making Based on Collision Risk Index. Master's Thesis, Dalian Maritime University, Dalian, China, 2015.
2. Qiao, S.; Shen, D.; Wang, X.; Han, N.; Zhu, W. A self-adaptive parameter selection trajectory prediction approach via hidden Markov models. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 284–296. [CrossRef]
3. Tso, S.K.; Liu, K.P. Demonstrated trajectory selection by hidden Markov model. In Proceedings of the 1997 IEEE International Conference on Robotics and Automation, Albuquerque, NM, USA, 20–25 April 1997; pp. 2713–2718. [CrossRef]
4. Cheng, Y.; Qiao, Y.; Yang, J. An improved Markov method for prediction of user mobility. In Proceedings of the 12th International Conference on Network and Service Management, Montreal, QC, Canada, 30 October–4 November 2016; pp. 394–399.
5. Fu, X.; Jiang, Y.; Lu, G.; Wang, J.; Huang, D.; Yao, D. Probabilistic trajectory prediction in intelligent driving. *IFAC Proc. Vol.* **2014**, *47*, 2664–2672. [CrossRef]
6. Lymperopoulos, I.; Lygeros, J. Adaptive aircraft trajectory prediction using particle filters. In Proceedings of the AIAA Guidance, Navigation and Control Conference and Exhibit, Honolulu, HI, USA, 18–21 August 2008. [CrossRef]
7. Xin, L.; Hailong, P.; Jianqiang, L. Trajectory prediction based on particle filter application in mobile robot system. In Proceedings of the 27th Chinese Control Conference, Kungming, Yunnan, China, 16–18 July 2008; pp. 389–394. [CrossRef]
8. Yang, S.Y.; Wu, T.; Zhang, Y.; Deng, F. Particle filter based on simulated annealing for target tracking. *J. Optoelectron Laser* **2011**, *22*, 1236–1240.
9. Perera, L.P.; Oliveira, P.; Soares, C.G. Maritime traffic monitoring based on vessel detection, tracking, state estimation, and trajectory prediction. *IEEE Trans. Intell. Transp. Syst.* **2012**, *13*, 1188–1200. [CrossRef]
10. Wang, L.; Zhang, L.; Yi, Z. Trajectory predictor by using recurrent neural networks in visual tracking. *IEEE Trans. Cybern.* **2017**, *47*, 3172–3183. [CrossRef] [PubMed]
11. Gao, M.; Shi, G.Y.; Li, S. Online prediction of ship behavior with automatic identification system sensor data using bidirectional long short-term memory recurrent neural network. *Sensors* **2018**, *18*, 4211. [CrossRef]
12. Ren, C.; An, N.; Wang, J.; Li, L.; Hu, B.; Shang, D. Optimal parameters selection for BP neural network based on particle swarm optimization: A case study of wind speed forecasting. *Knowl.-Based Syst.* **2014**, *56*, 226–239. [CrossRef]
13. Xiao, X.; Zhong, D.H.; Wang, D.; Lin, W.W.; Wang, J.J.; Liu, Z. Dynamic update of diversion tunnel construction simulation parameters based on ACDE-SVM. *J. Hydroelectr. Eng.* **2019**, *38*, 234–245. [CrossRef]
14. Cortes, C.; Vapnik, V. Support vector networks. *Mach. Learn.* **1995**, *20*, 273–295. [CrossRef]
15. Cherkassky, V.; Ma, Y. Practical selection of SVM parameters and noise estimation for SVM regression. *Neural Netw.* **2004**, *17*, 113–126. [CrossRef]
16. Vapnik, V. *The Nature of Statistical Learning Theory*; Springer: Berlin, Germany, 1995. [CrossRef]
17. Keerthi, S.S.; Gilbert, E.G. Convergence of a generalized SMO algorithm for SVM classifier design. *Mach. Learn.* **2002**, *46*, 351–360. [CrossRef]
18. Liu, Q.; Chen, C.; Zhang, Y.; Hu, Z.G. Feature selection for support vector machines with RBF kernel. *Artif. Intell. Rev.* **2011**, *36*, 99–115. [CrossRef]
19. Keerthi, S.S.; Lin, C.J. Asymptotic Behaviors of Support Vector Machines with Gaussian Kernel. *Neural Comput.* **2003**, *15*, 1667–1689. [CrossRef] [PubMed]
20. Abo-Khalil, A.G.; Lee, D.C. MPPT control of wind generation systems based on estimated wind speed using SVR. *IEEE Trans. Ind. Electron.* **2008**, *55*, 1489–1490. [CrossRef]
21. Hong, W.C. Electric load forecasting by seasonal recurrent SVR (support vector regression) with chaotic artificial bee colony algorithm. *Energy* **2011**, *36*, 5568–5578. [CrossRef]
22. Ko, C.N.; Lee, C.M. Short-term load forecasting using SVR (support vector regression)-based radial basis function neural network with dual extended Kalman filter. *Energy* **2013**, *49*, 413–422. [CrossRef]
23. Zhao, C.Y.; Zhang, H.X.; Zhang, X.Y.; Liu, M.C.; Hu, Z.D.; Fan, B.T. Application of support vector machine (SVM) for prediction toxic activity of different data sets. *Toxicology* **2006**, *217*, 105–119. [CrossRef]
24. Xiong, X.; Chen, L.; Liang, J. A new framework of vehicle collision prediction by combining SVM and HMM. *IEEE Trans. Intell. Transp. Syst.* **2017**, *19*, 399–710. [CrossRef]

25. Qin, A.K.; Huang, V.L.; Suganthan, P.N. Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Trans. Evol. Comput.* **2009**, *13*, 398–417. [CrossRef]
26. Ela, A.A.A.E.; Abido, M.A.; Spea, S.R. Optimal power flow using differential evolution algorithm. *Electr. Power Syst. Res.* **2010**, *80*, 878–885. [CrossRef]
27. Storn, R.; Price, K. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
28. Cai, H.R.; Chung, C.Y.; Wong, K.P. Application of differential evolution algorithm for transient stability constrained optimal power flow. *IEEE Trans. Power Syst.* **2008**, *23*, 719–728. [CrossRef]
29. Mayer, D.G.; Kinghorn, B.P.; Archer, A.A. Differential evolution an easy and efficient evolutionary algorithm for model optimisation. *Agric. Syst.* **2005**, *83*, 315–328. [CrossRef]
30. Das, S.; Mullick, S.S.; Suganthan, P.N. Recent advances in differential evolution–An updated survey. *Swarm Evol. Comput.* **2016**, *27*, 1–30. [CrossRef]
31. Li, X.; Yin, M. Parameter estimation for chaotic systems by hybrid differential evolution algorithm and artificial bee colony algorithm. *Nonlinear Dyn.* **2014**, *77*, 61–71. [CrossRef]
32. Coelho, L.S.; Mariani, V.C. Combining of chaotic differential evolution and quadratic programming for economic dispatch optimization with valve-point effect. *IEEE Trans. Power Syst.* **2006**, *21*, 989–996. [CrossRef]
33. Brest, J.; Zumer, V.; Maucec, M.S. Self-Adaptive Differential Evolution Algorithm in Constrained Real-Parameter Optimization. *IEEE Conf. Evol. Comput.* **2006**, 215–222. [CrossRef]
34. Guo, P. Research on Imrovement of Differential Evolution Algorithm. Master's Thesis, Tianjin University, Tianjin, China, 2011.
35. Lee, M.H.; Han, C.; Chang, K.S. Dynamic optimization of a continuous polymer reactor using a modified differential evolution algorithm. *Ind. Eng. Chem. Res.* **1999**, *38*, 4825–4831. [CrossRef]
36. Ho, W.H.; Chou, J.H.; Guo, C.Y. Parameter identification of chaotic systems using improved differential evolution algorithm. *Nonlinear Dyn.* **2010**, *61*, 29–41. [CrossRef]
37. Yuan, X.; Zhang, Y.; Wang, L.; Yuan, Y. An enhanced differential evolution algorithm for daily optimal hydro generation scheduling. *Comput. Math. Appl.* **2008**, *55*, 2458–2468. [CrossRef]
38. Zha, J.D. An approximation algorithm based on DE-SVR and its application in CPI forecast. *Bull. Sci. Technol.* **2012**, *28*, 39–41.
39. Piotrowski, A.P. Review of differential evolution population size. *Swarm Evol. Comput.* **2016**, 1–51. [CrossRef]
40. Mohamed, A.W.; Hadi, A.A.; Fattouh, A.M.; Jambi, K.M. LSHADE with semi-parameter adaptation hybrid with CMA-ES for solving CEC 2017 benchmark problems. *2017 IEEE Congr. Evol. Comput. (CEC)* **2017**, 145–152. [CrossRef]
41. Ronkkonen, J.; Kukkonen, S.; Price, K.V. Real-parameter optimization with differential evolution. *Evol. Comput.* **2005**, *1*, 506–513. [CrossRef]
42. Awad, N.H.; Ali, M.Z.; Liang, J.J.; Qu, B.Y.; Suganthan, P.N. *Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization*; Technical Report; Nanyang Technological University: Singapore; Jordan University of Science, Jordan and Technology and Zhengzhou University: Zhengzhou China, 2016.
43. Luo, Y.H. Ship Trajectory Prediction Based on AIS Data. Master's Thesis, South China University of Technology, Guangzhou, China, 2017.
44. Chang, C.C.; Lin, C.J. LIBSVM–A Library for Support Vector Machines. *ACM Tran. Intell. Sys. Technol.* **2011**, *2*, 1–39. [CrossRef]
45. Ma, Q.L.; Zheng, Q.L.; Peng, H.; Zhong, T.W.; Qin, J.W. Multi-step-prediction of chaotic time series based on co-evolutionary recurrent neural network. *Chin. Phys. B* **2008**, *17*, 536–542. [CrossRef]
46. Bitzer, S.; Kiebel, S.J. Recognizing recurrent neural networks (rRNN): Bayesian inference for recurrent neural networks. *Biol. Cybern.* **2012**, *106*, 201–217. [CrossRef]
47. Gupta, L.; McAvoy, M. Investigating the prediction capabilities of the simple recurrent neural network on real temporal sequences. *Pattern Recognit.* **2000**, *33*, 2075–2081. [CrossRef]