sensors

MDPI

Article

A Ship Trajectory Prediction Framework Based on a Recurrent Neural Network

Yongfeng Suo [1,*], Wenke Chen [1], Christophe Claramunt [2] and Shenhua Yang [1]

1 Navigation College, Jimei University, Xiamen 361021, China; 201811823002@jmu.edu.cn (W.C.); shyang@jmu.edu.cn (S.Y.)
2 Naval Academy Research Institute, BP 600, 29240 Brest Naval, France; christophe.claramunt@ecole-navale.fr
* Correspondence: yfsuo@jmu.edu.cn

Received: 2 August 2020; Accepted: 7 September 2020; Published: 9 September 2020

Abstract: Ship trajectory prediction is a key requisite for maritime navigation early warning and safety, but accuracy and computation efficiency are major issues still to be resolved. The research presented in this paper introduces a deep learning framework and a Gate Recurrent Unit (GRU) model to predict vessel trajectories. First, series of trajectories are extracted from Automatic Identification System (AIS) ship data (i.e., longitude, latitude, speed, and course). Secondly, main trajectories are derived by applying the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm. Next, a trajectory information correction algorithm is applied based on a symmetric segmented-path distance to eliminate the influence of a large number of redundant data and to optimize incoming trajectories. A recurrent neural network is applied to predict real-time ship trajectories and is successively trained. Ground truth data from AIS raw data in the port of Zhangzhou, China were used to train and verify the validity of the proposed model. Further comparison was made with the Long Short-Term Memory (LSTM) network. The experiments showed that the ship's trajectory prediction method can improve computational time efficiency even though the prediction accuracy is similar to that of LSTM.

Keywords: trajectory prediction; deep learning; DBSCAN; GRU; LSTM; redundant data

1. Introduction

AIS data provide real-time trajectory data which can be used to monitor ship's navigation status and trigger alert mechanisms for ship collision avoidance, maritime monitoring, trajectory clustering, ship traffic flow predicting, and maritime accident investigations [1]. The most basic information from AIS data gives spatiotemporal data composed of time and location, which is usually plotted on an electronic chart that can then form a ship trajectory.

In sea areas or ports with high traffic density and complicated conditions, a key issue is to improve the safety of ships sailing at sea. Vessel Traffic Service (VTS), whose objective is to accurately and effectively monitor and predict ship trajectories and real-time ship trajectories, provides a valuable technical support for early warning of marine traffic accidents [2]. To improve the safety of ships sailing in the environment of complex and changeable seas, it is necessary to provide trajectory prediction and danger warning functions to a ship's intelligent navigation system. However, the maritime navigation environment is prone to many incidents, especially in crowded port waters, and it is not easy to predict moving targets.

In a related work, a Bayesian probability trajectory prediction model based on a Gaussian process is introduced. A K-order multivariate Markov chain is applied to establish a state transition matrix to train a large amount of data and support short-term prediction of ship positions, but this approach is still sensitive to previous ship positions this resulting in low accuracy [3]. A single ship neighborhood approach based on historical data has also been suggested. The search algorithm can support short-term prediction, but it cannot deal with the branch trajectory problem [4].

The research presented in this paper introduces a deep learning theoretical framework and a Gate Recurrent Unit (GRU) model to predict vessel trajectories. The approach is based on an integration, clustering, and correction of maritime navigation trajectories. A recurrent neural network is applied to predict and train real-time ship trajectories. Ground truth data from AIS raw data in the port of Zhangzhou, China were used to train and verify the validity of the proposed model. Further comparison was made with the Long Short-Term Memory (LSTM) network that has been widely used in the field of deep learning. The rest of the paper is organized as follows. Section 2 briefly introduces related work and the motivation of our work. Section 3 develops the main principles behind our data preparation approach. Section 4 presents the GRU model. Section 5 reports on the experiments. Section 6 concludes the paper and outlines further work.

2. Related Work

Trajectory prediction methods based on statistical methods are commonly used in the ship field, including Gaussian process regression models, which are the most common. Anderson [5] took time as the independent variable, obtained the measured value in discrete time, and regarded the trajectory as a one-dimensional Gaussian process. The method defines the prior continuous time through the nonlinear time-varying stochastic differential equation driven by white noise, calculates the posterior distribution of the predicted value by obtaining the joint prior density and covariance matrix of the observed value and the predicted value, and uses dynamics in combination, odel smooth trajectory estimation. Rong [6] decomposed ship motion into horizontal and vertical. In the lateral direction, a Gaussian process is used to model the uncertainty of lateral motion, and the longitudinal direction is estimated by acceleration. The horizontal distribution model obtains the hyperparameters of the Gaussian regression model through historical AIS data, and the mean function and covariance function are determined by the hyperparameters. The predicted trajectory can be estimated by evaluating the mean value and covariance matrix, that is, the mean value and variance are used to describe the ship's lateral position and its uncertainty. The advantage of Gaussian process regression is its strong applicability and easy to understand. The disadvantage is that the amount of calculation is large, and as the forecasting time passes, the accuracy of the forecast results will be greatly reduced. Jiang [7] used a polynomial Kalman filter to fit the piecewise polynomial features of the ship's track. Kalman Filter (KF) can estimate the state of the system under the condition of uncertain factors. It combines joint distributions at different times to estimate unknown variables. This method first uses polynomial fitting to obtain the state equation and observation equation; determines the initial state, error covariance matrix, and other parameters; and then it completes the first step of filtering. Next, it performs N-step prediction and compares the predicted value with the true value. The algorithm iterates N times to update the error covariance matrix of the track points to complete the trajectory prediction.

The Kalman filter implements this algorithm in the form of recursion. Its advantage is that it occupies less storage space during the calculation process and can realize short-term trajectory prediction. The disadvantage is that the initial state of the model and the assumptions under ideal conditions have a significant impact on the accuracy of prediction.

Neural networks are computing systems with interconnected nodes that work similar to neurons in the human brain. Using algorithms, they can recognize hidden patterns and correlations in raw data, cluster and classify them, and—over time—continuously learn and improve [8]. Neural networks are also ideally suited to solving complex problems in real-life situations. They can learn and model the relationships between inputs and outputs that are nonlinear and complex; make generalizations and inferences; reveal hidden relationships, patterns, and predictions; and model highly volatile data (such as time series data) [9]. With the popularization of artificial intelligence, neural networks have also been gradually applied to the field of maritime navigation [10,11]. Historical ship trajectory data and trajectory characteristics are used as an input, while predicted ship trajectory data are the output of the neural network [11]. Zhang et al. proposed a deep learning method that integrates multiple ship movements, which can be adapted to predict different many categories of ship trajectories after training the neural network appropriately [12]. Overall, the resulting accuracy varies as a function of ship categories, thus the modeling approach still has to be improved.

Brian [13] proposed a dual linear autoencoder method to predict the future trajectory of ships. Autoencoder (AE) includes two processes of encoding and decoding, which can extract hidden features in the data and reconstruct the original input data with new features. The autoencoder can send the extracted new features to the supervised learning model and then use them to predict the trajectory. This model first clusters ship trajectories based on historical AIS data to predict the trajectories of ships in the selected category. This method generates the entire ship trajectory, rather than the iterative state based on historical predictions. Through the potential distribution of possible future trajectories of ships, the model can predict multiple ship trajectories and predict their uncertainties. The autoencoder can extract deep-level data features. In addition, using sparse autoencoder can obtain better initial parameters in engineering applications. However, the extraction of deep-level data features requires accurate grasp. Excessive extraction will lead to the extraction of useless data features, making the model effect poor.

Mao [14] proposed an ELM-based marine trajectory prediction method to predict the trajectory of ships. Extreme Learning Machine (ELM) is a machine learning algorithm based on a single hidden layer feedforward neural network. Since the ELM algorithm does not require the weights and biases of the iterative neural network, the training time of the ELM neural network is less than that of the traditional neural network. The advantage of this algorithm is that the generalization performance of the model is better, and the calculation speed can be improved. However, it is more dependent on the number of model nodes and the selection of some optimal parameters.

A "sequence-to-sequence" recurrent neural network model has been developed to mesh and serialize a ship trajectory into a neural network model, in order to predict the main trajectory and arrival time [15]. A LSTM model has been introduced to predict ship's position by evaluating the probability distribution and obtains relatively valid results [16]. To improve the accuracy of the prediction mechanisms, a multiple azimuth autonomous device sensor has been used as an additional data input, but the approach relies on a large amount of AIS data so computational performance is relatively low [17]. While large AIS historical data can be used as references for predicting maritime trajectories, data quality is often not guaranteed, and data redundancy is a major problem: worldwide there are about 1600 AIS receivers on the coastline of more than 150 countries and 65,000 ships sailing around the world [18]. Furthermore, abnormal data due to either environmental conditions or technical issues are likely to generate significant trajectory prediction errors [19,20]. The most appropriate balance between the necessary training of the prediction mechanisms with large enough trajectory data, and the negative impact of redundant and noisy data is still a crucial issue for practicality [21]. The advantages of common statistics-based methods are that the data occupy less storage space during the calculation process, can realize short-term trajectory prediction, and the calculation method is relatively lightweight, achieving an effect similar to deep learning. The disadvantage is that the initial state of the model and the assumptions under ideal conditions have a large impact on the prediction results. However, statistical methods cannot learn the effects of shallow reefs, islands, and other spatial factors on the trajectory of ships as deep learning. This feature makes deep learning more practical in trajectory prediction. This motivates our search for a trajectory prediction approach that will take into account the impact of redundant and noisy data on the neural network training and optimize the input trajectory dataset to improve the final quality of the modeling approach. This leads us to combine a neural network with a GRU framework and that is introduced in the following sections.

3. Modeling Approach

The prediction model framework consists of four parts: data preprocessing, clustering analysis, similarity measurement, and model analysis (Figure 1). Data preprocessing is an essential part of deep learning, as cleaned data can improve the model performance. Clustering analysis uses historical data to extract the main ship trajectory areas. Similarity evaluation is based on the symmetric segmented-path distance to eliminate the influence of a large number of redundant data. The model analysis applies the deep learning model of a lightweight recurrent neural network GRU to train the model and predict ship trajectories.

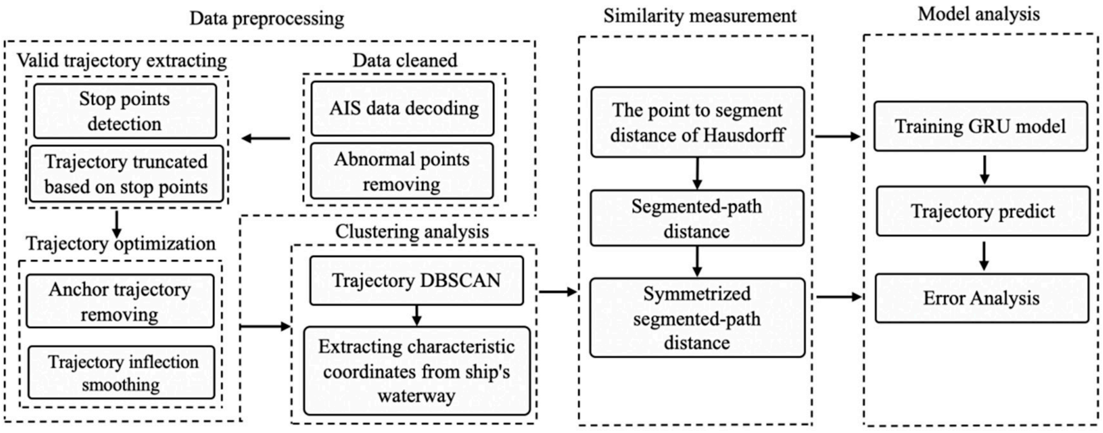

Data preprocessing
Valid trajectory extracting
Stop points detection
Trajectory truncated based on stop points
Data cleaned
AIS data decoding
Abnormal points removing
Trajectory optimization
Anchor trajectory removing
Trajectory inflection smoothing
Clustering analysis
Trajectory DBSCAN
Extracting characteristic coordinates from ship's waterway
Similarity measurement
The point to segment distance of Hausdorff
Segmented-path distance
Symmetrized segmented-path distance
Model analysis
Training GRU model
Trajectory predict
Error Analysis

Figure 1. Flowchart of the vessel trajectory prediction framework.

3.1. Data Preprocessing

The AIS is an essential component of modern vessel navigation systems, which are installed and widely used on vessels to enhance the ability to identify targets and mark the location. Raw AIS data are used to build a reference dataset and define a matrix of historic AIS data as follows:

$$X_{T_j} = [X_1\ ,\ X_2,\ \cdots\ ,\ X_N]^T \tag{1}$$

where N is the total number of AIS messages and

$$X_i = \left[MMSI_i\ ,\ t_i\ ,\ {p_i}^T,\ c_i\ ,v_i\right] \tag{2}$$

$i\ \in\ \{1,\ 2,\ldots,N\}$ is a vector where $MMSI_i$, t_i, ${p_i}^T$, and v_i represent, respectively, the Maritime Mobile Service Identity (MMSI), timestamp, location (WGS84 longitude and latitude), course over ground (COG), and speed over ground (SOG). The MMSI is a unique identifier for each vessel.

During the data preprocessing process, the wandering or anchoring trajectories in the original dataset are eliminated. We set the minimum time interval of the trajectory to 1200 s, because of the AIS information receiving interval is generally specified to be about 5–10 min, and the information interval higher than 20 min is used as the next stage of navigation status [16]. Each trajectory is optimized by a function Optimization (V_i) as shown in Algorithm 1. Figures 2 and 3 show a comparison of two trajectories' density heat maps before and after preprocessing. The green dots in Figure 2 show the location of the vessel, the red region indicates dense areas, and many error data or anchor trajectories are still contained. Figure 3 shows that green dots after processing are smooth and in sailing state, and yellow to red dots indicate denser trajectories. Many noisy data in the original data are eliminated. In the original ship data, the dense red ring area shown in Figure 2 contains data about many ships floating and anchoring. The speed of these ships is affected by wind and ocean currents and is often less than 1 knot, which is an abnormal navigation state. The data required for the experiment are the data of the ship in a sailing state; accordingly, a route performs smoothly along the motion trajectory, which makes each trajectory easier to analyze.

```
Algorithm 1: Data preprocessing
Input: Original dataset $X_{T_j}$, minimum voyage time interval threshold $\varepsilon$.
Output: Trajectories $T_{s_i}$.
1: Connect to the database.
2: Get $V_i$ where $i$ in $\mathrm{Len}(O)$ and $V_i$.//$\Delta t$ is the total time interval of voyage $i$.
3: if $R_i.mmsi = R_{i-1}.mmsi$ &&$R_i.sog < 1kn$ && $\delta_t < 1200s$ where $\delta_t = R_i.t - R_{i-1}.t$//$R_i$ is the next state of
   $V_i$, $R_i.sog$ is the speed of the vessel, $\delta_t$ is the time interval between two-state.
4: then Optimization $(V_i)$, $T_{s_i} \rightarrow V_i$ .//Optimize $(\cdot)$ is the trajectory optimizing function.
Return: $T_{s_i}$
```

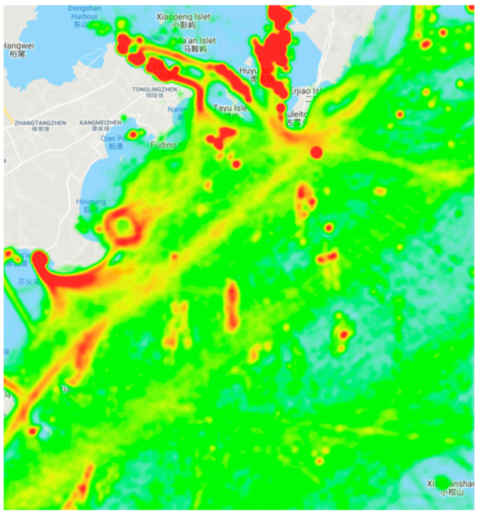


Figure 2. A visualization of the original AIS data in Zhangzhou, Fujian, China. The green line indicates the trajectory, and darker color indicates greater trajectory density. For example, the red area indicates the area with a large trajectory density.

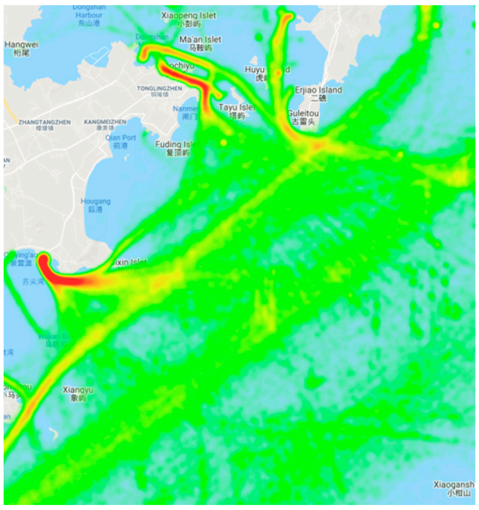


Figure 3. This is the heat map of the trajectory data processed by the algorithm; the trajectory in the state of drifting, anchoring, etc. is cleared, and the trajectory data in the normal sailing state is saved.

In Table 1, the total number of coordinates after preprocessing is reduced by nearly 71.8% compared to the initial data, and the number of trajectories is reduced by almost 92.1%, as only the vessels underway are saved as valid trajectories (e.g., anchor or wander is filtered). Nearly all noisy data have been eliminated from raw data, and more valid and regular vessel trajectories are obtained, which can improve the processing efficiency.

Table 1. Comparison before and after data processing.

Data	Total Coordinates (Points)	Number of Ships	Number of Voyages
Raw Data	7,577,484	13,437	1,004,137
After Data Preprocessing	2,134,324	7231	78,531

During the actual reception and sending of AIS information in navigation state, due to the phenomenon of signal drift and artificial tampering of the AIS equipment, the received ship information is likely to show some abnormal locations, as illustrated in Figure 4. As shown in Figure 4a, the speed of the ship in the dataset is roughly concentrated at about 20 knot but also includes speed below 1 knot. In Figure 4b, the dataset contains some abnormal speeds greater than 20 knot. To restore the true state of the navigation, AIS information should be optimized to a valid trajectory which is approximated by

$$P_{T_j} = \left[lat_{T_j}, lon_{T_j}, V_{T_j}, \theta_{T_j}, \omega_{T_j}\right] \tag{3}$$

while the average speed of the ship during a period can be estimated by

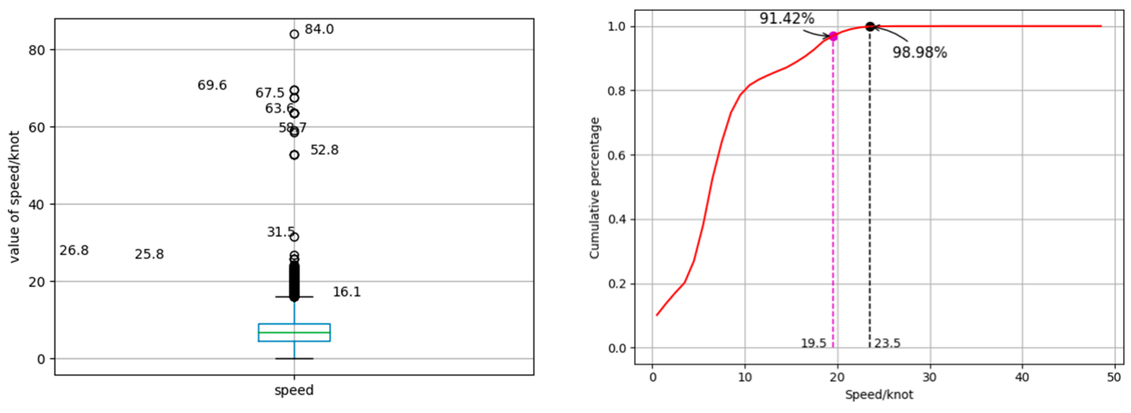
$$\widetilde{V}_{T_j} = \frac{\sqrt{\left(lat_{T_j} - lat_{T_{j+1}}\right)^2 + \left(lon_{T_j} - lon_{T_{j+1}}\right)^2}}{T_{j+1} - T_j} \tag{4}$$

where lat_{m,T_j} and lon_{m,T_j} are the longitude and latitude coordinates of the ship m at time T_j. At this time, the shipping speed should meet the condition given by

$$V_{T_j} - a'\left(T_{j+1} - T_j\right) \leq V_{T_j} \leq V_{T_j} - a\left(T_{j+1} - T_j\right) \tag{5}$$

where a and a' are the forward and reverse acceleration when the ship is moving forward. When the current data are identified as an abnormal speed, the average speed of the two points adjacent to the point is substituted to the data.

$$V_{T_j} = \begin{cases} V_{T_j}, & \textit{normal speed} \\ \widetilde{V}_{T_j}, & \textit{abnormal speed} \end{cases} \tag{6}$$



(**a**) (**b**)

Figure 4. The two forms of AIS data ship speed's distribution picture: (**a**) box diagram of ship speed, from which it can be seen that the speed is mainly distributed around 15, and a speed greater than 60 is an abnormal speed; and (**b**) probability distribution diagram, where normal data accounts for 98.98% of the total data.

When judging whether the data are tested as an inflection point or a sudden change point, we can approximate the location by

$$\begin{cases} \hat{lat}_{T_j} = lat_{T_j} + V_{\hat{T}_j}\left(T_{j+1} - T_j\right) \\ \hat{lon}_{T_j} = lon_{T_j} + V_{\hat{T}_j}\left(T_{j+1} - T_j\right) \end{cases} \tag{7}$$

where lat_{m,T_j} and lon_{m,T_j} are the position of the ship at time T_j. $\hat{lat}_{m,T_j}$ and $\hat{lon}_{m,T_j}$ give the location of the ship at time T_{j+1}. If the current location of the ship does not satisfy Equation (8), the current position information is replaced as follows:

$$\sqrt{\left(lat_{T_j} - lat_{T_{j+1}}\right)^2 + \left(lon_{T_j} - lon_{T_{j+1}}\right)^2} \le 0.5a\left(T_{j+1} - T_j\right)^2 \tag{8}$$

Due to the non-equal interval of AIS data reception, when the data update interval is too low, the ship's speed and acceleration are used to interpolate the course of the route following [9]. Figure 5 shows a comparison of two trajectories before and after optimization.

$$lat_{T_j} = lat_{T_{j-1}} + \frac{1}{2} \cdot \Delta t \cdot \left[V_{T_j} + \Delta t \cdot a_{T_{j-1}} \cdot \cos\left(\frac{\pi\theta}{180} + \Delta t \cdot \omega_{T_{j-1}}\right)\right] \tag{9}$$

$$lon_{T_j} = lon_{T_{j-1}} + \frac{1}{2} \cdot \Delta t \cdot \left[V_{T_j} + \Delta t \cdot a_{T_{j-1}} \cdot \sin\left(\frac{\pi\theta}{180} + \Delta t \cdot \omega_{T_{j-1}}\right)\right] \tag{10}$$

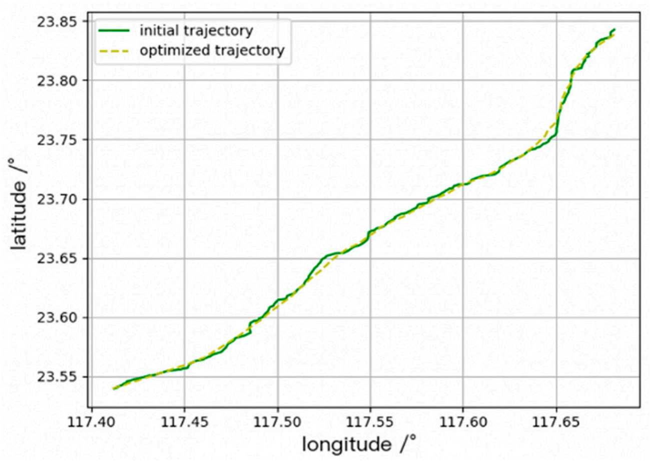


Figure 5. Plot of the comparison of the original trajectory and optimized trajectory of a same vessel.

3.2. Cluster Analysis

This section introduces the trajectory clustering model based on the AIS data that were previously processed to generate regular shipping route patterns.

The DBSCAN algorithm is a density-based point or line clustering algorithm that was introduced by Ester et al. [22]. The main advantage of this algorithm is that arbitrarily shaped clusters can be identified. By applying the DBSCAN algorithm, lines are mainly divided into three types: density-connected lines, outliers, and core lines. A line is generally classified as a core line if a minimum number of MinPts lines are included within a distance Eps, and the lines that are density connected with others are regarded as the same cluster, while points which are not density connected are regarded as outliers [23].

The optimized trajectory set is generated by the DBSCAN trajectory clustering, and trajectories are roughly classified into incoming waterway, crossing waterway, and north–south routes. This allows us to extract the area of each trajectory to be predicted, and then each classified waterway is used as a dataset separately for subsequent data processing, as shown on Figure 6.

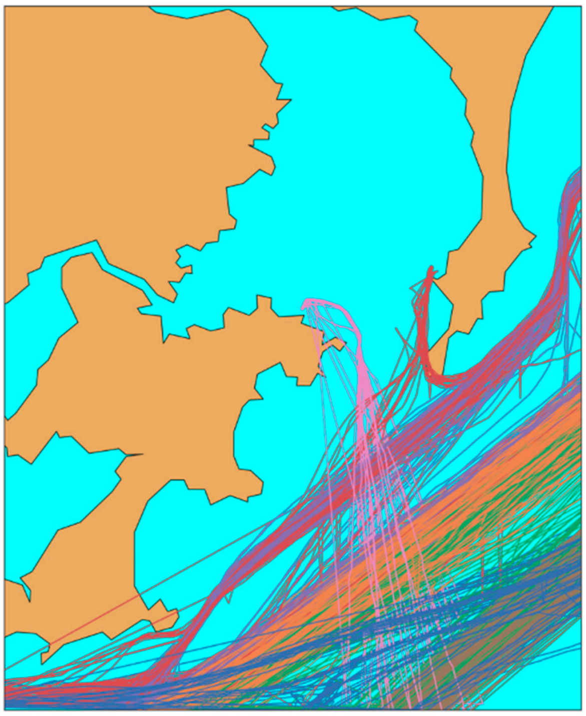


Figure 6. Trajectory clusters by DBSCAN. Each color cluster represents a different densest waterway.

This section uses statistical inferences on the crossing points of multiple ship trajectories through the cross section and extracts the trajectory with the characteristics of the trajectory cluster as the similarity measurement trajectory as shown in Figure 7. The specific process is as follows:

Step 1: Select a segment of clustered trajectory cluster data, classify the AIS trajectory point set according to MMSI, and connect the AIS trajectory points into a polyline with speed and direction.

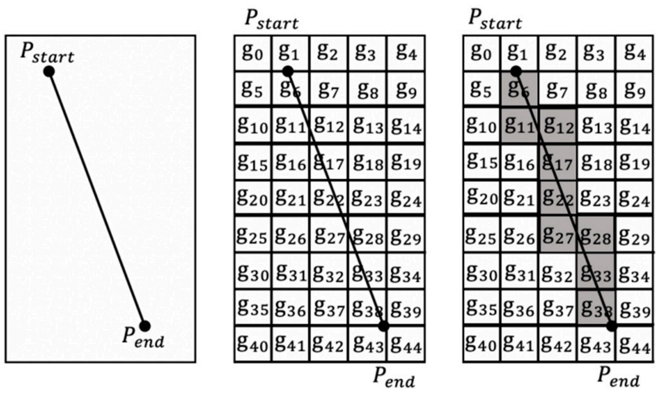
Step 2: Select two coordinate points on the ship trajectory cluster to form a line as the cross section, and grid the surrounding position information into $g_0, g_1, \cdots, g_n$, to record the crossing position of the ship trajectory.

Step 3: Use the line and line intersection algorithm to determine whether the ship's trajectory passes through the cross section, count the number of crossing points the ship passes through the cross section, and store it in the grid id.

Step 4: The frequency at which the ship traverses the position is expressed in the form of probability density. The estimated value of the grid unit uses the Gaussian kernel function. The grid of the longitude and latitude coordinates corresponding to the peak of the probability density curve is selected as the Get used as crossing areas as shown in Figure 8.

Step 5: Intercept multiple times on the track clusters of the ship trajectory in this segment, repeat Step 4 to get the grid id that the ship is accustomed to traverse, until a trajectory is obtained that fully displays the features of the track in this segment.

Step 6: Use the most representative trajectory obtained in Step 5 as the similarity measurement trajectory. The specific similarity measurement method is introduced in Section 3.3.



P_{start}
P_{end}
g_0 g_1 g_2 g_3 g_4
g_5 g_6 g_7 g_8 g_9
g_{10} g_{11} g_{12} g_{13} g_{14}
g_{15} g_{16} g_{17} g_{18} g_{19}
g_{20} g_{21} g_{22} g_{23} g_{24}
g_{25} g_{26} g_{27} g_{28} g_{29}
g_{30} g_{31} g_{32} g_{33} g_{34}
g_{35} g_{36} g_{37} g_{38} g_{39}
g_{40} g_{41} g_{42} g_{43} g_{44}

Figure 7. After gridding, the shaded area represents the grid for counting the number of times the ship has passed through the cross section.

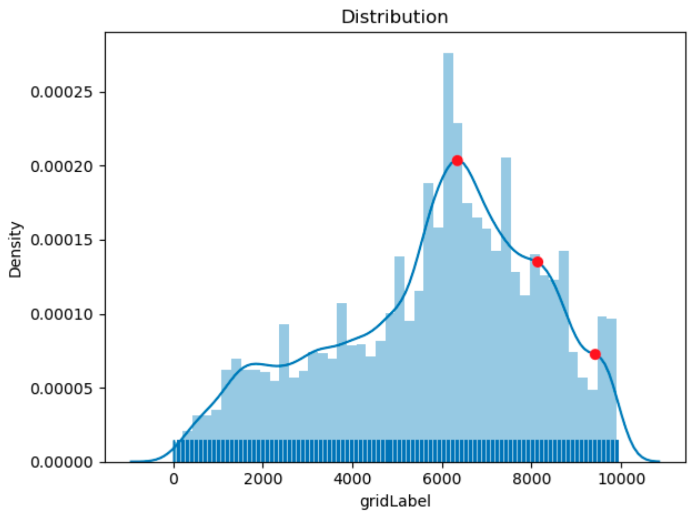


Figure 8. The gridded area shows the number of crossings recorded by each grid in the form of frequency.

3.3. Trajectory Similarity Measurement

The quality of the incoming data has an important impact on vessel trajectory predictions. Due to the large difference in the shape of each trajectory, as shown in Figure 9, there are many redundant data that are not related to the target trajectory. To solve this problem, there is a need to obtain more reliable and useful datasets. A data preprocessing algorithm based on the Symmetrized Segment-Path Distance (SSPD) is applied [24]: First, let the historical trajectory after data preprocessing and the target trajectory be evaluated by calculating the similarity coefficient between each trajectory. Second, a filtering process is triggered according to the similarity coefficient. Finally, a subset of relevant trajectory data that meet the conditions is obtained. The SSPD model is described in detail below.

Most vessel trajectories are curves or straight, and trajectory similarity measurements mainly depend on the following conditions:

Nearly similarity shape
Physically closer to each other
Similar as a whole, rather than just similar subparts

The definition of the SSPD is based on a point-to-segment distance, which is derived by the Hausdorff distance, D_{pt}, as shown in Figure 10. It is given by the minimum of distances between the points of all segments. The segmented-path distance from the trajectory T^1 to the trajectory T^2 is given by the average of all distances from each point that compose T^1 to the trajectory T^2, as shown in Figure 11.

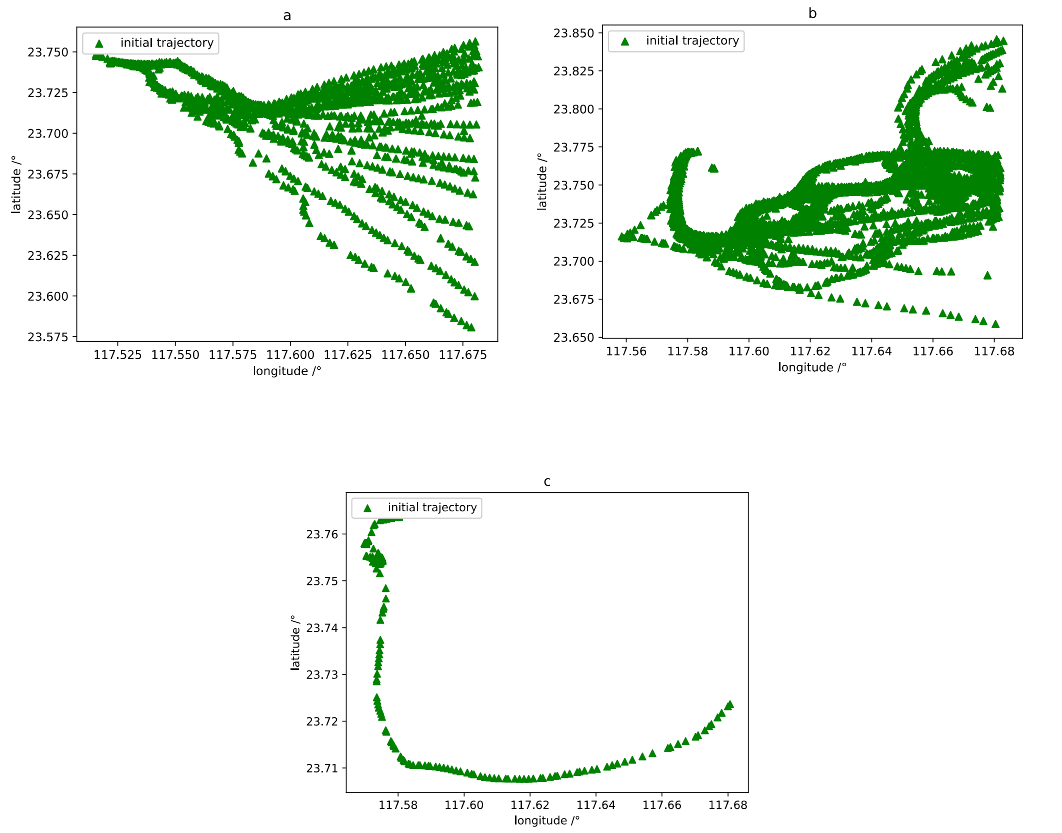


Figure 9. The three representative common vessel trajectories: (**a**) straight trajectory entering a port; (**b**) circuitous trajectory in narrow seas; and (**c**) trajectory with large turns in wide seas.

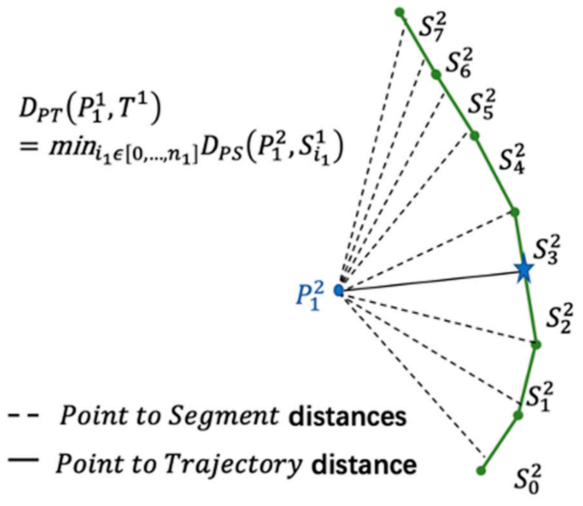

$D_{PT}(P_1^1, T^1)$
$= min_{i_1 \epsilon [0,\ldots,n_1]} D_{PS}(P_1^2, S_{i_1}^1)$
S_7^2
S_6^2
S_5^2
S_4^2
S_3^2
P_1^2
S_2^2
S_1^2
S_0^2
-- *Point to Segment* **distances**
— *Point to Trajectory* **distance**

Figure 10. Distance from point p_1^2 to T^1. S_i^2 represents segments. The distance of P_1^2 between S_3^2 is the minimum.

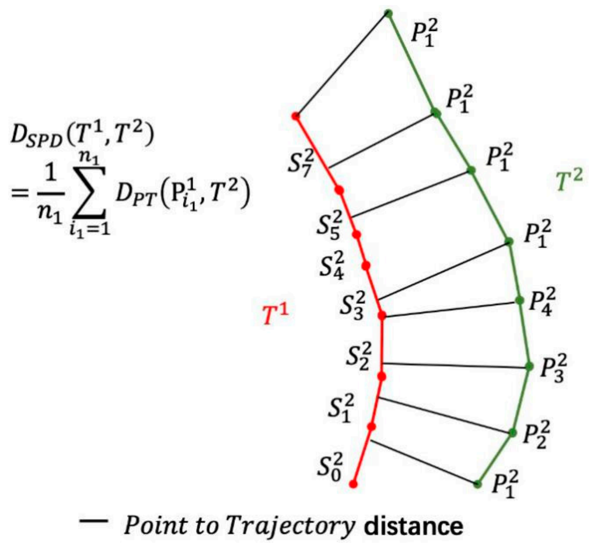


Figure 11. SPD distance from the trajectory T^1 to the trajectory T^2.

Suppose one of the targets to be measured is $T_s{}^1$, and the other known trajectory is $T_s{}^2$. Since the SSPD algorithm is proposed based on the Hausdorff distance, according to the definition of the midpoint of the Hausdorff distance to the line segment, the distance between the line segment on the trajectory $T_s{}^2$ and the trajectory $T_s{}^1$ is given as

$$D_{ps}\left(p_{i_1}^1, s_{i_2}^2\right) = \begin{cases} \|p_{i_1}^2 p_{i_2}^{2proj}\|_2\,, & \text{if } p_{i_2}^{2proj} \epsilon\, s_{i_1}^1 \\ \min\left(\|p_{i_2}^2 p_{i_1}^1\|_2, \|p_{i_2}^2 p_{i_1+1}^1\|_2\right), & \text{otherwise} \end{cases} \tag{11}$$

where $p_{i_2}^{iproj}$ represents the orthogonal projection of the point p_k^i on the trajectory segment s_l^j and $\|p_{i_2}^2 p_{i_1}^1\|_2$ represents the Euclidean distance between $p_{i_2}^2$ and $p_{i_1}^1$.

The distance from the point on the trajectory $T_s{}^2$ to the trajectory $T_s{}^1$ is defined as

$$D_{pt}\left(p_{i_1}^1, T_s{}^1\right) = \min_{i_1 \epsilon [0,\ldots,n_1]} D_{ps}\left(p_{i_2}^2, s_{i_1}^1\right) \tag{12}$$

The segmented-path distance from the trajectory $T_s{}^2$ to the trajectory $T_s{}^1$ is defined as

$$D_{SPD}\left(T^1, T^2\right) = \frac{1}{n_1}\sum_{i_1=1}^{n_1} D_{pt}\left(p_{i_1}^1, T^2\right) \tag{13}$$

The symmetric segmented-path distance from the trajectory $T_s{}^2$ to the trajectory $T_s{}^1$ is based on the segmented-path distance from the trajectory $T_s{}^2$ to the trajectory $T_s{}^1$ which can be written as

$$D_{SSPD}\left(T^1, T^2\right) = \frac{D_{SPD}\left(T^1, T^2\right) + D_{SPD}\left(T^2, T^1\right)}{2} \tag{14}$$

The smaller is the value of $D_{SSPD}\left(T^1, T^2\right)$, the higher is the degree of similarity between $T_s{}^1$ and $T_s{}^2$.

Let the trajectory made of the grid processed as in Section 3.2 be target trajectory and compare the similarity with these trajectories $[T_{s_1}\ T_{s_2}\ \cdots\ T_n]^T$. The data preprocessing flow based on SSPD is shown in Algorithm 2. The similarity value of each known trajectory and the target trajectory is calculated by the SSPD algorithm; the similarity threshold is set according to the actual situation; and, among all similarity coefficients, the trajectory within the threshold is selected and used as the dataset.

Algorithm 2: Trajectory Similarity Measurement

```
Input: Trajectory dataset $Tr_i$ preprocessed by Algorithm 1, similarity threshold $\varepsilon$.
Output: Trajectories $T_{sr_i}$.
1: for $T_{s_i}$ in $[T_{s_1}\ T_{s_2}\ \cdots\ T_n]^T$
2: calculate the distance $D_{SSPD_i}(T_{s_i}, T_s)$ between $T_{s_i}$ and $T_s$
3: end for
4: for $D_{SSPD_i}(T_{s_i}, T_s)$ in $\left[D_{SSPD_1}\ D_{SSPD_2}\ \cdots\ D_{SSPD_n}\right]$
5: save the trajectory which $D_{SSPD_i} < \varepsilon$
6: end for
Return: $[T_{sr_1}\ T_{sr_2}\ \cdots\ T_{sr_m}]$
```

4. GRU Model

A Recurrent Neural Network (RNN) is a type of recurrent neural network that takes sequence data as an input while cyclic units are connected in a chain [25] and that can deal with short-term prediction problems. However, it has been observed that RNN is prone to the problem of gradient disappearance when training the network. This leads us to introduce the GRU model initially introduced by Cho et al. [26] to improve the performance of an RNN network. The GRU structure contains the hidden layer state of the original RNN and is also a variant of LSTM. The GRU structure is similar to the LSTM model but computationally cheaper, as the GRU combines the forgotten gate and the input gate of the LSTM into a single update gate. The main advantage is that the GRU structure can store historical time-series information. It uses a gating mechanism to remember as long as possible previous trajectories, while simplifying the processing. It has a distinct hierarchical structure different from LSTM in the sense that GRU has fewer parameters than LSTM, but it can perform similarly to LSTM. The main principles behind the GRU structure is shown in Figure 12.

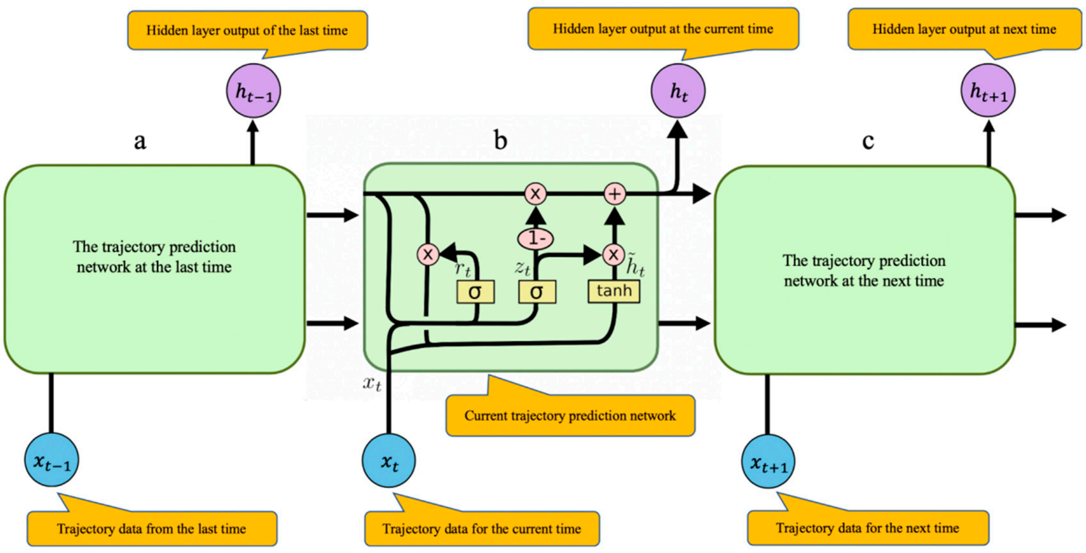

Hidden layer output of the last time
Hidden layer output at the current time
Hidden layer output at next time
h_{t-1}
h_t
h_{t+1}
a
b
c
The trajectory prediction network at the last time
The trajectory prediction network at the next time
r_t
z_t
$\tilde{h}_t$
σ
σ
tanh
1-
x_t
Current trajectory prediction network
x_{t-1}
x_t
x_{t+1}
Trajectory data from the last time
Trajectory data for the current time
Trajectory data for the next time

Figure 12. Schematic diagram of the trajectory prediction model based on the GRU neural network: (**a**) GRU unit at the last time; (**b**) GRU unit of the current time, which describes the detailed information of the transfer process; and (**c**) GRU unit at the next time.

Let us introduce more formally the principles of the GRU neural network. For the hidden layer of the GRU neural network, given the input value $x_t (t = 1, 2, \cdots, n)$, the value of the hidden layer at t is [27]:

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \tag{15}$$

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \tag{16}$$

$$\widetilde{h}_t = \tanh\left(W_{\widetilde{h}} \cdot [r_t * h_{t-1}, x_t]\right) \tag{17}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \widetilde{h}_t \tag{18}$$

where σ is sigmoid $(x) = \frac{1}{1+e^{-x}}$, $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$, the calculation method [ ] indicates that two vectors are connected, $*$ means multiplying matrix elements, and $\cdot$ indicates matrix multiplications as follows:

$$\boldsymbol{m} * \boldsymbol{n} = \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ \vdots \\ m_n \end{bmatrix} * \begin{bmatrix} n_1 \\ n_2 \\ n_3 \\ \vdots \\ n_n \end{bmatrix} = \begin{bmatrix} m_1 n_1 \\ m_2 n_2 \\ m_3 n_3 \\ \vdots \\ m_n n_n \end{bmatrix} \tag{19}$$

Equations (9)–(12) show that the weight matrices at time t that the GRU neural network needs to train are W_z, W_r, and $W_{\widetilde{h}}$, which are, respectively, composed of two weight matrices as follows:

$$W_z = W_{zx} + W_{z\widetilde{h}} \tag{20}$$

$$W_r = W_{rx} + W_{r\widetilde{h}} \tag{21}$$

$$W_{\widetilde{h}} = W_{\widetilde{h}x} + W_{\widetilde{h}\widetilde{h}} \tag{22}$$

where W_{zx}, W_{rx}, and $W_{\widetilde{h}x}$ are the weight matrix from the input value to the update gate, the reset gate, and the candidate value, respectively. $W_{z\widetilde{h}}$, $W_{r\widetilde{h}}$, and $W_{\widetilde{h}\widetilde{h}}$ are the weight matrix from the last candidate value to the update gate z_t, the reset gate r_t, and the candidate value $\widetilde{h}$ respectively.

"1-" indicates that each element in the vector will be subtracted from 1. For the value of z_t at time t, the larger is the value, the less the hidden layer value h_t at time $t-1$ is affected by the hidden layer value h_t at time $t-1$, and the more affected it is by the candidate value $\widetilde{h}_t$ at time t. If the value of z_t is close to 1, it means that the value h_{t-1} of the hidden layer at $t-1$ does not contribute to the value h_t of the hidden layer at time t. z_t can better reflect the impact of the data with longer time intervals to the current moment in the time series sequence. The larger is the value r_t at time t, the greater is the influence of the candidate value $\widetilde{h}_t$ at time t on the hidden layer value h_{t-1} at time $t-1$. If the value of r_t is approximately null, it means that the hidden layer value h_{t-1} at time $t-1$ does not contribute to the candidate value $\widetilde{h}_t$ at time t, and r_t can better reflect the impact of the data with short time intervals to the current time in the time series sequence. The GRU neural network hidden layer structure is shown in Figure 13.

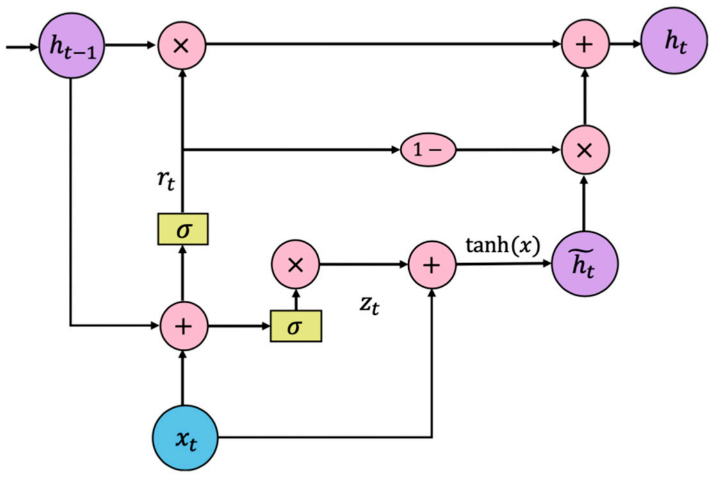


Figure 13. Schematic diagram of the GRU neural network hidden layer structure.

GRU merges the forgotten gate layer f_t and input gate i_t of the LSTM unit into the z_t of the GRU unit, as well as merges the hidden state and unit state of the LSTM network, together with the r_t, which is used to subject the degree of ignoring the status information of the previous time to control the flow of ship trajectory information. The updated gate is used to control how much status information of the previous moment is brought into the current state. Specifically, it combines the newly input trajectory information of the next time and the previous memory to determine what information is used in the current state to calculate the state information of the next time. The combination of GRU threshold structures retains the most important data to ensure that some information is not lost during long-term propagation. Due to the relatively simple structure of the GRU model, fewer parameters need to be trained, and it has the advantage of fast training speed during the training process.

5. Experiments and Analysis

The vessel trajectory datasets are classified after DBSCAN clustering, using the clusters of the north–south routes as the experimental dataset, which is divided into three parts: the training set, validation set, and test set. The training set is used to train the model. In the training process, the validation set is used to verify the performance of the model to improve its generalization ability [28]. The test set is used to generate some prediction results according to the input and output ways of N to 1 where the status vectors of the first N steps are taken as the input sequence and the status of the next time of the vessel as the output vectors. Then, the state of the first N minutes of a vessel is the original trajectory and the state of the vessel after N minutes is the one to be predicted. To ensure dimensionless interference between the data, the original time series is converted into the input sequence of the recurrent neural network after being normalized by

$$X_{std} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{23}$$

where X_{std} is the normalized data and x_{min} is the minimum value of the sample. After normalization, the data are mapped in the interval $[0,\ 1]$.

The result of output processed by the neural network in the interval $[0, 1]$ is denormalized and mapped to the original dimension level of the sample by

$$X_{scaler} = x_{std}(x_{max} - x_{min}) + x_{min} \tag{24}$$

where X_{scaler} is the data after denormalization.

To evaluate the results of GRU in vessel traffic flow prediction, LSTM is constructed as comparative experiments, and Mean Square Error (MSE) is selected as the model error analysis index during the training process as followed.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{25}$$

where y_i is the ground-truth trajectory value and; $\hat{y}_i$ is the predicted value.

The selection of parameters is crucial for recurrent neural networks. Generally speaking, the more hidden layers are included in the GRU recurrent neural network, the stronger is the model's performance and learning ability. The Adam optimizer [29], which is a stochastic gradient descent and adaptive learning rate optimization method, was used in the experiment. It can avoid the training process from falling into a local optimum situation. After determining the sampling rate as 1 in the experiment, the essential parameters in the network, batch size and the number of neurons, were experimentally compared and demonstrated to obtain the best parameter combination [30]. The selectable range of the batch size was {8, 16, 24, 32}, and the range of the number of neurons could be selected from {60, 80, 100, 120, 140, 160}.

The selection of parameters such as the neurons is shown in Tables 2 and 3. As the number of batch size gradually increases, the calculation time gradually decreases. When the number of batch sizes is too small or large, the generated errors are excessive. When the quantity of neurons is 16 and 24, respectively, the calculation time is similar, and the former error is smaller, thus a valuable number of batch size is given as 16. As shown in Table 3, as the number of neurons gradually increases, the consumption time also gradually increases. When the number of neurons increases from 60 to 120, the prediction error gradually decreases. When the number of neurons increases further, the error increases, thus, for the neurons, it is more reasonable to set the number at 100.

Table 2. Comparison of the experimental influence of different parameters of batch size on the two models.

Model	Metrics	Number of Batch Size			
		8	16	24	32
GRU	MSE	1.38×10^{-3}	1.03×10^{-3}	1.47×10^{-3}	2.87×10^{-3}
	Time Resuming/s	205.23	124.03	108.00	87.14
LSTM	MSE	1.26×10^{-3}	1.03×10^{-3}	1.46×10^{-3}	3.03×10^{-3}
	Time Resuming/s	221.24	128.32	103.21	90.34

Table 3. Comparison of the experimental influence of different parameters of neurons on the two models.

Model	Metrics	Number of Neurons					
		60	80	100	120	140	160
GRU	MSE	6.72×10^{-3}	3.01×10^{-3}	1.93×10^{-3}	1.02×10^{-3}	4.59×10^{-3}	4.63×10^{-3}
	Time Resuming/s	86.99	103.43	124.03	135.97	154.53	176.74
LSTM	MSE	6.34×10^{-3}	2.89×10^{-3}	1.93×10^{-3}	1.02×10^{-3}	4.03×10^{-3}	4.27×10^{-3}
	Time Resuming/s	88.43	107.34	124.59	133.62	153.43	173.75

After several experimental tests and parameter comparisons, the GRU construct consists of the following parts: fully connected layer, an activation layer, two GRU layers, and a method of drop out to construct the GRU recurrent neural network. The second layer gru_1 and fourth layer gru_2 are the GRU layers, each containing 120 hidden units; the third and fifth layers are the drop out method, which can not only ensure that the model maintains the robustness of the model when information is lost during training, but it can also be used for regularization, reducing the weight connection, increasing the robustness of the network model in the event of missing individual connection information; and the fifth layer is a fully connected layer, which contains two neurons. The visualization of the GRU neural network is shown in Figure 14.

In the training process, as shown in Figure 14, both models iterate quickly, and the training loss value can quickly obtain a better convergence effect. When the number of training rounds is about 30, GRU has reached the extremum; LSTM as a control test reached the extremum value at 50 rounds. The improved GRU model based on LSTM can converge to the extreme point more quickly than the LSTM model in the training phase. The fast convergence speed is mainly due to the simplified GRU gate structure for the gate design of the LSTM structure.

The model training is shown in Figures 15 and 16, and one of the results of the trajectory fitting process is presented in Figure 17, from which we can see that the fitting effect is satisfactory. When the training phase reaches about 30 rounds, the accuracy of the GRU model reaches 96%; the accuracy rate from the 200th round to the 300th round reaches about 98%; and then it gradually smooths, which shows that the efficiency of GRU is better than that of LSTM.

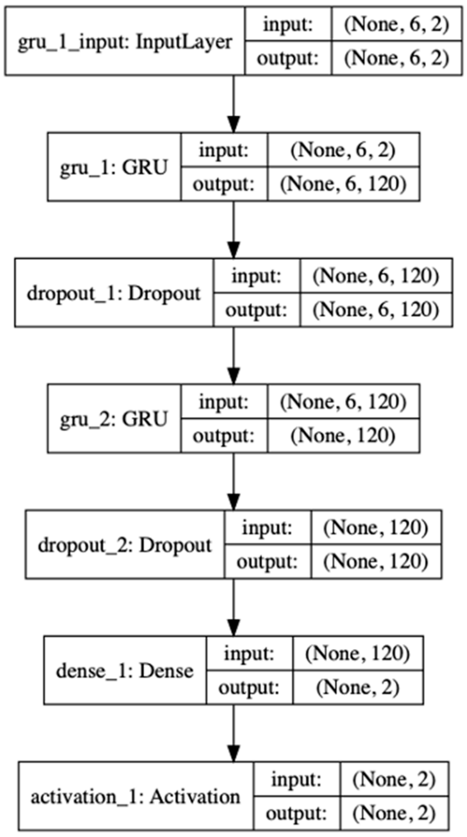


Figure 14. Plot of all parameters setting of GRU recurrent neural network model.

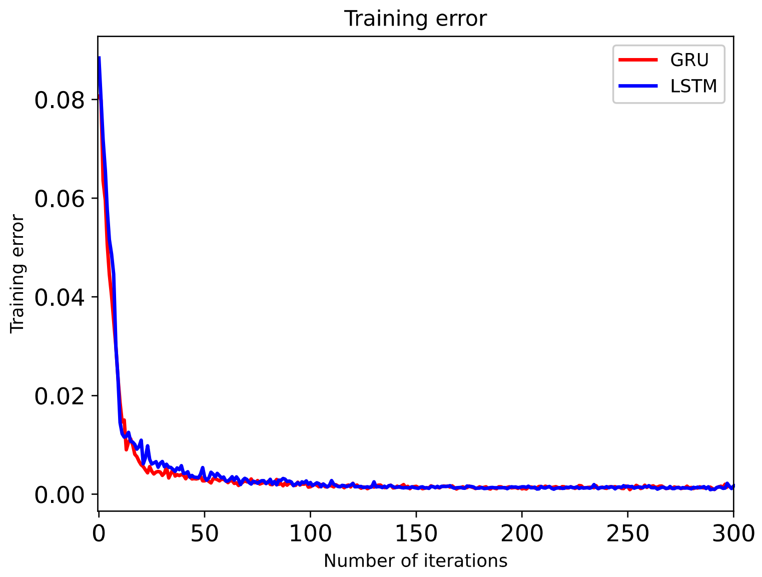


Figure 15. Plot of comparison of iterative convergence of two models.

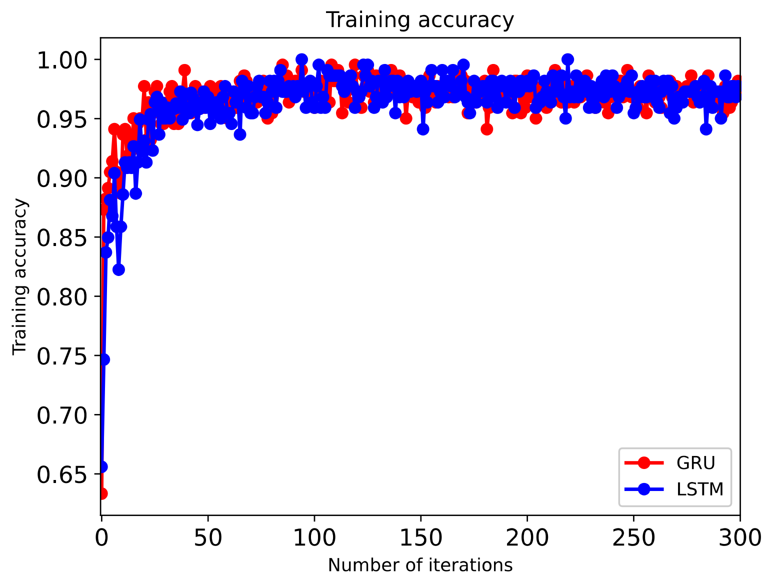


Figure 16. Plot of comparison of prediction accuracy diagram of two models.

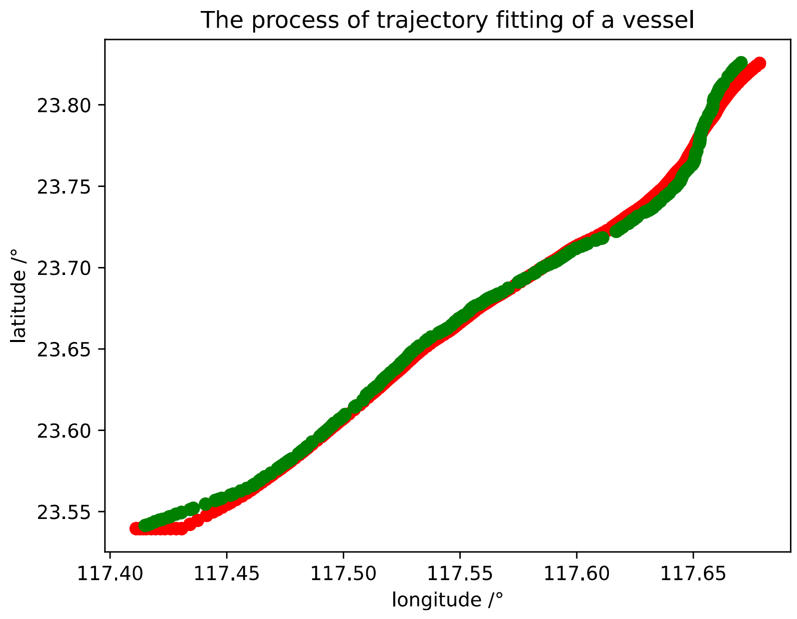


Figure 17. Plot of comparison of a real trajectory and fitting trajectory uses the GRU model.

A total of 330 sets of trajectory data was used to predicted using LSTM, Extended Kalman Filter (EKF), and GRU models as shown in Figure 18. From the experimental results, the deep learning method is better than the traditional statistical-based method EKF; EKF's prediction results are good, but, compared with deep learning, the prediction error is not stable and much bigger errors appear due to the strong nonlinearity of the data. Based on the same framework, the prediction results of GRU and LSTM are relatively similar, and the errors are basically stable within 0.02. With similar prediction accuracy, it can be seen in Tables 2 and 3 that the calculation efficiency of GRU model is slightly better than LSTM.

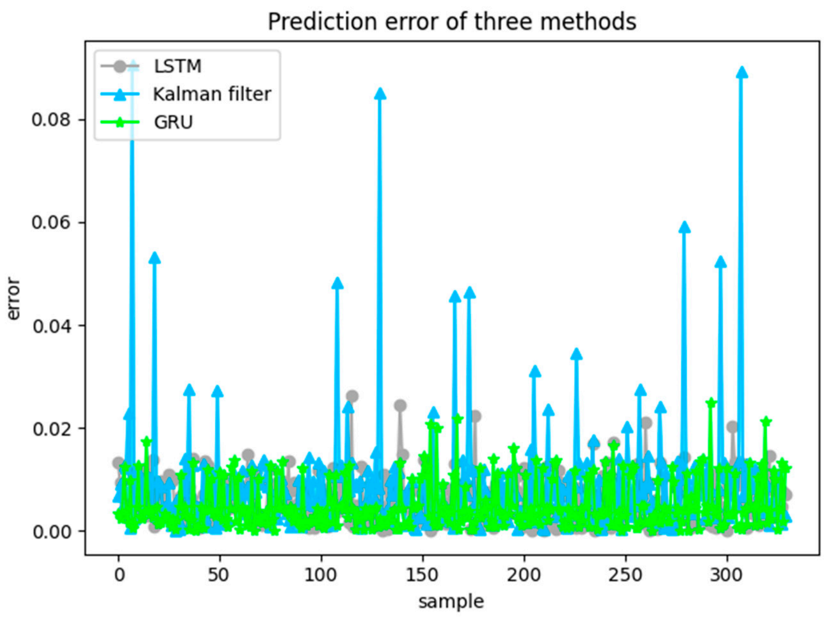


Figure 18. Prediction errors of the three methods calculated by equation (25): gray solid circle markers indicate the prediction results of LSTM, blue solid triangle indicate those of EKF, and the green solid stars markers denote those of GRU model.

A vessel trajectory is selected from the historical AIS data of Zhangzhou Port, China, visualized in the Electronic Chart Display and Information System (ECDIS), as shown in Figures 19 and 20. The prediction trajectory of the GRU model is illustrated, and prediction results are visualized. The gray line represents the observed history trajectory until the current time, the red line denotes the predicted trajectories, the yellow line represents the ground-truth trajectory at the 30th minute of the path, while the green line represents the course vector bar of the vessel. There is a turning point in the trajectory, which can verify that the predictive performance of our model can be applied to nonlinear data. The predicted trajectory almost overlaps with the real trajectory, and a good prediction effect is achieved.

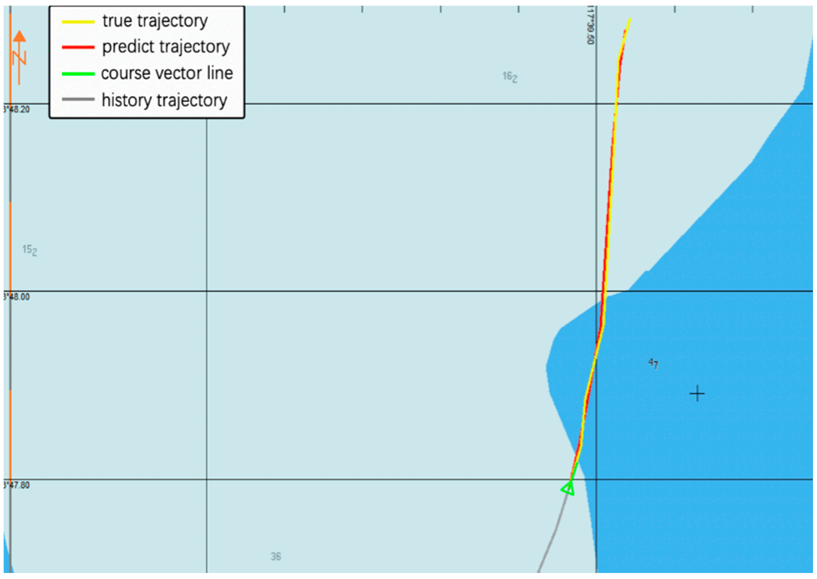


Figure 19. Visualization of the vessel’s real trajectory and predicted trajectory that uses the GRU model.

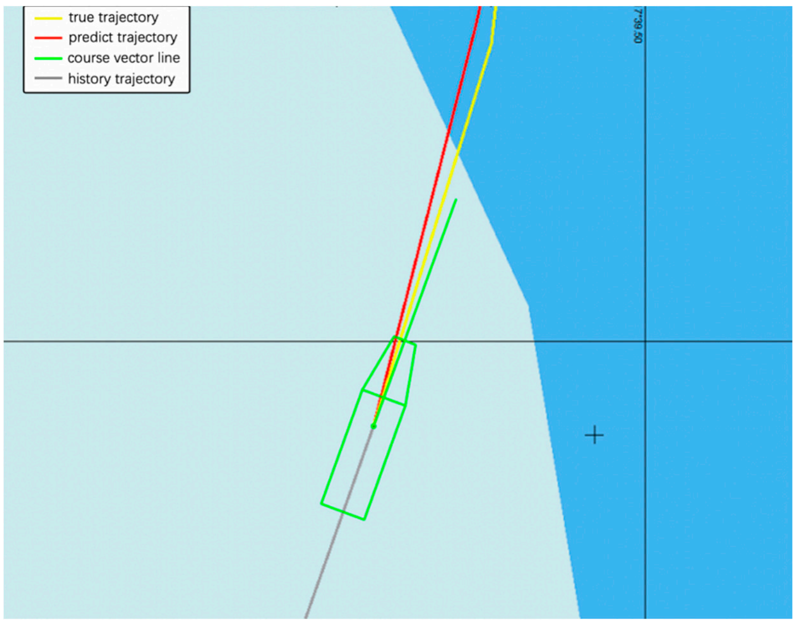


Figure 20. A detailed representation of the vessel's real trajectory and predicted trajectory that uses the GRU model.

6. Conclusions

This paper introduces a ship trajectory prediction framework based on a recurrent neural network. One of the peculiarities of our approach is to take into account vessel navigation errors due to receiving and sending data processes. The vessel trajectory prediction model is divided into data preprocessing, clustering analysis, similarity measurement, and model analysis. The objective of the first three modules is to extract high-quality datasets. The model was tested with a series of real data and different parameter combinations within a reasonable range o train the neural network and predict vessel trajectories is the most valuable way. A comparative experiment was conducted with LSTM that is considered as a valuable solution. The experiments showed that the vessel trajectory prediction model based on the GRU model has good prediction accuracy similar to that of LSTM, but it has the advantage of being computationally more efficient. It is more suitable for the requirements of immediate and early warnings of maritime navigation and can provide appropriate decisions for intelligent vessel navigation systems and VTS. LSTM is similar to GRU, and it is less effective in long-distance trajectory prediction. On the basis of improving the calculation efficiency, if the accuracy of long-distance trajectory prediction can be improved, it will be of great help to the early warning of ship navigation hazards. While current trajectory prediction model is based on a recursive call model, long-distance vessel trajectories are still computationally expensive (e.g., predicting the vessel's arrival at the port), which will be considered in further work.

Author Contributions: Conceptualization, Y.S. and W.C.; methodology, Y.S.; software, W.C.; validation, Y.S. and C.C.; formal analysis, S.Y.; investigation, S.Y.; resources, S.Y.; data curation, S.Y.; writing—original draft preparation, W.C.; writing—review and editing, Y.S. and C.C.; visualization, S.Y.; supervision, C.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China under Grants 51579114 and 51879119 and in part by the Natural Science Foundation of Fujian Province under Grant 2018J01536.

Acknowledgments: The authors wish to express thanks to the Lina Li and Guoquan Chen for their helpful preliminary work and experiment cooperation. The authors also thank the reviewers for valuable comments and suggestions.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. 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]
2. Praetorius, G. Vessel Traffic Service (VTS): A Maritime Information Service or Traffic Control System? Understanding Everyday Performance and Resilience in a Socio-technical System Under Change. Ph.D. Thesis, Chalmers University of Technology, Göteborg, Sweden, 2014.
3. Guo, S.; Liu, C.; Guo, Z.; Feng, Y.; Hong, F.; Huang, H. Trajectory prediction for ocean vessels base on K-order multivariate Markov chain. In Proceedings of the International Conference on Wireless Algorithms, Tianjin, China, 20–22 June 2018; pp. 140–150.
4. Hexeberg, S.; Flåten, A.L.; Eriksen, B.H.; Brekke, E. AIS-based vessel trajectory prediction. In Proceedings of the 2017 20th International Conference on Information Fusion (Fusion), Xi'an, China, 10–13 July 2017; pp. 1–8.
5. Anderson, S.; Barfoot, T.D.; Tong, C.H.; Särkkä, S. Batch nonlinear continuous-time trajectory estimation as exactly sparse Gaussian process regression. *Auton. Robot* **2015**, *39*, 221–238. [CrossRef]
6. Rong, H.; Teixeira, A.P.; Soares, C.G. Ship trajectory uncertainty prediction based on a Gaussian Process model. *Ocean Eng.* **2019**, *182*, 499–511. [CrossRef]
7. Jiang, B.; Guan, J.; Zhou, W.; Chen, X. Vessel trajectory prediction algorithm based on polynomial fitting kalman filtering. *J. Signal Process.* **2019**, *35*, 741–746. [CrossRef]
8. Abiodun, E.O.; Jantan, A.; Omolara, A.E.; Dada, K.V.; Mohamed, N.; Arshad, H. State-of-the-art in artificial neural network applications: A survey. *Heliyon* **2018**, *4*, e938. [CrossRef]
9. Zaremba, W.; Sutskever, I.; Vinyals, O. Recurrent neural network regularization. *arXiv* **2014**, arXiv:1409.2329.
10. De Vries, G.K.D.; van Someren, M. Machine learning for vessel trajectories using compression, alignments and domain knowledge. *Expert Syst. Appl.* **2012**, *39*, 13426–13439. [CrossRef]
11. Zhong, C.; Jiang, Z.; Chu, X.; Liu, L. Inland ship trajectory restoration by recurrent neural network. *J. Navig.* **2019**, *72*, 1359–1377. [CrossRef]
12. Tu, E.; Zhang, G.; Mao, S.; Rachmawati, L.; Huang, G.-B. Modeling historical AIS data for vessel path prediction: A comprehensive treatment. *arXiv* **2020**, arXiv:abs/2001.01592.
13. Murray, B.; Perera, L.P. A dual linear autoencoder approach for vessel trajectory prediction using historical AIS data. *Ocean Eng.* **2020**, *209*, 107478. [CrossRef]
14. Mao, S.; Tu, E.; Zhang, G.; Rachmawati, L.; Rajabally, E.; Huang, G.-B. An automatic identification system (AIS)database for maritime trajectory prediction and data mining. In *Proceedings of ELM-2016*; Springer: Cham, Switzerland, 2018; pp. 241–257.
15. Nguyen, D.-D.; Le Van, C.; Ali, M.I. Vessel trajectory prediction using sequence-to-sequence models over spatial grid[C]. In Proceedings of the 12th ACM International Conference on Distributed and Event-Based Systems, Hamilton, New Zealand, 25–29 June 2018; Association for Computing Machinery: New York, NY, USA, 2018; pp. 258–261.
16. Tang, H.; Yin, Y.; Shen, H. A model for vessel trajectory prediction based on long short-term memory neural network. *J. Mar. Eng. Technol.* **2019**, 1–10. [CrossRef]
17. Borkowski, P. The ship movement trajectory prediction algorithm using navigational data fusion. *Sensors* **2017**, *17*, 1432. [CrossRef]
18. Zissis, D.; Xidias, E.K.; Lekkas, D. Real-time vessel behavior prediction. *Evol. Syst.* **2016**, *7*, 29–40. [CrossRef]
19. Lei, P.R.; Su, I.J.; Peng, W.C.; Han, W.Y.; Chang, C.P. A framework of moving behavior modeling in the maritime surveillance. *Chung Cheng Ling Hsueh Pao/J. Chung Cheng Inst. Technol.* **2011**, *40*, 33–44.
20. Pallotta, G.; Vespe, M.; Bryan, K. Vessel pattern knowledge discovery from AIS data: A framework for anomaly detection and route prediction. *Entropy* **2013**, *15*, 2218–2245. [CrossRef]
21. Valsamis, A.; Tserpes, K.; Zissis, D.; Anagnostopoulos, D.; Varvarigou, T. Employing traditional machine learning algorithms for big data streams analysis: The case of object trajectory prediction. *J. Syst. Softw.* **2017**, *127*, 249–257. [CrossRef]
22. Ester, M.; Kriegel, H.P.; Sander, J.R.; Xu, X. A density-waBased algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Portland, OR, USA, 2–4 August 1996; AAAI Press: Menlo Park, CA, USA, 1996; pp. 226–231.
23. Li, H.; Liu, J.; Wu, K.; Yang, Z.; Liu, R.W.; Xiong, N. Spatio-temporal vessel trajectory clustering based on data mapping and density. *IEEE Access* **2018**, *6*, 58939–58954. [CrossRef]

24. Besse, P.C.; Guillouet, B.; Loubes, J.-M.; Royer, F. Review and perspective for distance-based clustering of vehicle trajectories. In *IEEE Transactions on Intelligent Transportation Systems*; IEEE: Piscataway, NJ, USA, 2016; Volume 17, pp. 3306–3317.
25. Hochreiter, S.; Schmidhuber, J.U.R. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
26. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv* **2014**, arXiv:1412.3555.
27. Jozefowicz, R.; Zaremba, W.; Sutskever, I. An empirical exploration of recurrent network architectures. In Proceedings of the 32nd International Conference on International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 2342–2350.
28. Sun, L.; Zhou, W. Vessel motion statistical learning based on stored ais data and its application to trajectory prediction. In Proceedings of the 2017 5th International Conference on Machinery, Materials and Computing Technology (ICMMCT 2017), Beijing, China, 25–26 March 2017; pp. 1183–1189.
29. Kingma, D.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980v9, 2014.
30. Hu, Y.K.; Xia, W.; Hu, X.X.; Sun, H.Q.; Wang, Y.H. Vessel trajectory prediction based on recurrent neural network. *Syst. Eng. Electron.* **2020**, *42*, 871–877.