Journal of
Marine Science and Engineering

Article

Informer-Based Model for Long-Term Ship Trajectory Prediction

Caiquan Xiong, Hao Shi, Jiaming Li, Xinyun Wu * and Rong Gao

School of Computer Science, Hubei University of Technology, Wuhan 430068, China; xiongcq@hbut.edu.cn (C.X.); 102211152@hbut.edu.cn (H.S.); lijiaming@hbut.edu.cn (J.L.); gaorong@hbut.edu.cn (R.G.)
* Correspondence: xinyun@hbut.edu.cn

Abstract: Ship trajectory prediction is a complex time series forecasting problem that necessitates models capable of accurately capturing both long-term trends and short-term fluctuations in vessel movements. While existing deep learning models excel in short-term predictions, they struggle with long-sequence time series forecasting (LSTF) due to difficulties in capturing long-term dependencies, resulting in significant prediction errors. This paper proposes the Informer-TP method, leveraging Automatic Identification System (AIS) data and based on the Informer model, to enhance the ability to capture long-term dependencies, thereby improving the accuracy of long-term ship trajectory predictions. Firstly, AIS data are preprocessed and divided into trajectory segments. Secondly, the time series is separated from the trajectory data in each segment and input into the model. The Informer model is utilized to improve long-term ship trajectory prediction ability, and the output mechanism is adjusted to enable predictions for each segment. Finally, the proposed model's effectiveness is validated through comparisons with baseline models, and the influence of various sequence lengths L_{token} on the Informer-TP model is explored. Experimental results show that compared with other models, the proposed model exhibits the lowest Mean Squared Error, Mean Absolute Error, and Haversine distance in long-term forecasting, demonstrating that the model can effectively capture long-term dependencies in the trajectories, thereby improving the accuracy of long-term vessel trajectory predictions. This provides an effective and feasible method for ensuring ship navigation safety and advancing intelligent shipping.

Keywords: Informer; trajectory prediction; long-sequence forecasting; trajectory planning

check for updates

Citation: Xiong, C.; Shi, H.; Li, J.; Wu, X.; Gao, R. Informer-Based Model for Long-Term Ship Trajectory Prediction. *J. Mar. Sci. Eng.* **2024**, *12*, 1269. https://doi.org/10.3390/jmse12081269

Academic Editor: Rafael Morales

Received: 7 July 2024
Revised: 23 July 2024
Accepted: 26 July 2024
Published: 28 July 2024

1. Introduction

With the rapid development of the global shipping industry, the number of ships is steadily increasing, and maritime traffic has become more complex. The risk of maritime traffic accidents is rising, posing significant threats to human life and causing severe environmental damage. Thus, implementing effective measures to enhance maritime traffic safety is imperative. The Vessel Traffic Service (VTS) is a system designed to monitor and manage maritime traffic, tracking ship movements and providing navigation safety guidance within a limited range [1]. This system effectively enhances maritime safety and efficiency, reduces the incidence of accidents, and aids in marine environmental protection. Moreover, it offers navigation assistance, providing critical data support and real-time monitoring for ship trajectory prediction, thereby greatly improving prediction accuracy and reliability. However, in congested port areas, high traffic volume and frequently changing conditions still lead to various traffic accidents. To meet these challenges, it is essential to develop more robust trajectory prediction capabilities for ship navigation systems and trajectory planning, thereby improving the safety of ships in complex and dynamic environments.

The development of the AIS [2] has significantly advanced maritime vessel identification and navigation technologies, providing crucial support for ship trajectory prediction and collision warning systems. AIS-based ship trajectory prediction enables the planning of safer and more efficient navigation routes. However, during daily operations, various external factors can introduce noise into the AIS data, such as measurement errors and sensor malfunctions. This noise often leads to issues like duplicate data, discontinuities, and errors, which severely impact prediction accuracy [3]. Consequently, preprocessing the collected data to remove noise and enhance data quality is a common practice. Research on AIS data involves statistical analysis with large datasets [4,5], the application of linear regression models [6], and probabilistic models [7], addressing the redundancy, sparsity, and anomalies in AIS data to achieve more accurate trajectory predictions.

Numerous scholars have researched ship trajectory prediction, primarily using machine learning and deep learning approaches. Xue et al. [8] introduced a model using Gaussian processes and moment-matching methods, achieving high accuracy by addressing prediction uncertainty. Zhang et al. [9] developed a model based on random forests to predict destinations by comparing current and historical trajectories. Liu et al. [10] presented a Support Vector Regression (SVR) model optimized with the Adaptive Chaotic Differential Evolution algorithm (ACDE), enhancing convergence speed and prediction accuracy. Zhou et al. [11] proposed a method using AIS data and BP neural networks, excelling in short-term predictions but limited in medium- and long-term scenarios.

Advancements in deep neural networks have enhanced ship trajectory prediction methods, improving the ability to learn navigation patterns and capture trajectory changes. Most models in this field are studied based on Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs). Zhang et al. [12] combined LSTM with Generative Adversarial Networks (GANs) to handle AIS data with varying time intervals, bypassing the need for uniform preprocessing. Park et al. [13] used DBSCAN to remove outliers from ship trajectories and employed Bi-LSTM for higher accuracy compared to LSTM. Fu et al. [14] applied XGBoost to ConvLSTM to extract both temporal and spatial features. Han et al. [15] developed a GRU-based method for high accuracy in short-term predictions. Additionally, incorporating attention mechanisms into neural networks has further improved prediction performance [16–18].

At the same time, combining various base models for trajectory sequence prediction has also shown high accuracy. Wu et al. [19] integrated ConvLSTM with Seq2Seq to capture spatiotemporal features, improving accuracy for both turning and linear trajectories. Similarly, combining LSTM with Seq2Seq [20] and incorporating attention mechanisms [21] has also yielded good results. Lin et al. [22] introduced a hybrid model, TTCN-Attention-GRU (TTAG), by combining Temporal Convolutional Networks (TCNs), attention mechanisms, and GRU networks, achieving high accuracy. Chen et al. [23] developed a method using forward GRU and backward BiGRU networks, employing bidirectional scaling to enhance trajectory fitting accuracy.

In summary, in the current field of ship trajectory sequence prediction, RNNs are effective in considering the dependencies between adjacent data points over time, while CNNs can capture similarities at different positions. These capabilities are well suited for trajectory prediction tasks. However, existing deep learning models have the following issues: (1) Most deep learning methods focus on short-term ship trajectory prediction, with limited research on long-term predictions. Long-term ship trajectory prediction can provide crucial information for navigation path planning, aiding in the development of safer and more efficient routes, thus holding significant application value. (2) When handling long-sequence time predictions, models tend to have larger parameter sizes, longer training times, and reduced operational efficiency.

The Transformer model [24] differs from traditional RNN and CNN models, achieving significant breakthroughs in natural language processing (NLP) [25] and offering new approaches and methods for other machine learning tasks [26]. In the realm of time series prediction, the traditional Transformer model has limitations [27], prompting many scholars to enhance the Transformer model to better address time series prediction challenges. For instance, Informer [28] improves time efficiency and storage efficiency in long-sequence prediction through an innovative self-attention mechanism and Decoder design. Autoformer [29] introduces a self-correlation mechanism based on sequence periodicity, which enhances prediction accuracy and efficiency. FEDformer [30] combines seasonal-trend decomposition with Transformers and incorporates frequency domain analysis to significantly improve long-term time series prediction performance. Pyraformer [31] introduces the pyramidal attention module (PAM), which can effectively describe short-term and long-term temporal dependencies with low time and space complexity. PatchTST [32] significantly surpasses existing Transformer-based models in long-term prediction accuracy by segmenting the time series data and ensuring channel independence.

In the task of ship trajectory sequence prediction, many scholars have made various improvements to the Transformer model to enhance accuracy in both short-term and long-term predictions. Dong et al. [33] proposed an attention mechanism model based on position encoding that extracts temporal correlations in trajectories by quantifying ship movements, achieving high accuracy in long-sequence predictions. Jiang et al. [34] incorporated the LSTM structure into the Transformer framework and proposed a trajectory time window shifting and smoothing filtering method to handle outliers in trajectory data, ensuring the continuity and integrity of trajectory points and thus improving prediction accuracy. Zhao et al. [35] proposed an improved DLinear model that identifies temporal and spatial dependencies and patterns in sequence data using a retrospective window, leading to enhanced training efficiency and improved short-term and long-term prediction accuracy. Nguyen et al. [36] introduced a new discrete, high-dimensional representation of AIS data and the improved Transformer network TrAISformer, which can extract long-term correlations in AIS trajectories to predict ship positions in the upcoming hours. Qiang et al. [37] proposed MSTFormer, a model that significantly enhances long-term trajectory prediction accuracy by integrating dynamic knowledge and spatial-temporal features, particularly when dealing with complex situations such as ship turns.

Inspired by the success of Informer in LSTF, this study investigates the application of the Informer model to ship trajectory prediction and proposes the Informer-TP model for this purpose. This paper presents a method for denoising AIS data and dividing them into trajectory segments to predict future trajectories. Leveraging the Informer architecture, this method effectively predicts long-term trajectories, with notable advantages in learning long-term dependencies and handling data efficiently. The contributions of this paper are as follows: (1) Denoising, clustering, and interpolating trajectory data to obtain highly correlated trajectories, which are then segmented into independent trajectory segments based on input and output steps. (2) Developing the Informer-TP model using these trajectory segments to predict ship trajectories accurately. (3) Examining the effect of different sequence lengths on the Informer-TP model. This study aims to aid trajectory planning, enhance ship trajectory prediction accuracy, prevent collisions, and improve maritime traffic safety and navigation efficiency.

The structure of the remaining sections is as follows: Section 2 details the prediction methods. Section 3 describes the data preprocessing and compares the performance of different models to analyze the effect of various sequence lengths on the model. Section 4 discusses the results of the proposed model. Section 5 concludes the study.

2. Methodology

In this section, we introduce the method of using the Informer-TP model to predict ship trajectories. The main framework of the proposed method is shown in Figure 1. First, we filter the AIS data to remove anomalies. Then, we separate and cluster the trajectory data to eliminate outliers. Next, we interpolate the trajectory, divide it into segments, resample, and normalize it to obtain the training data. Finally, we input the data into the model's Encoder. The encoding layer transforms it into the input representation, which the Encoder then processes. We feed a segment of real data and the masked data to be predicted into the Decoder for encoding. The Decoder concatenates this with the Encoder's output and computes the prediction results.

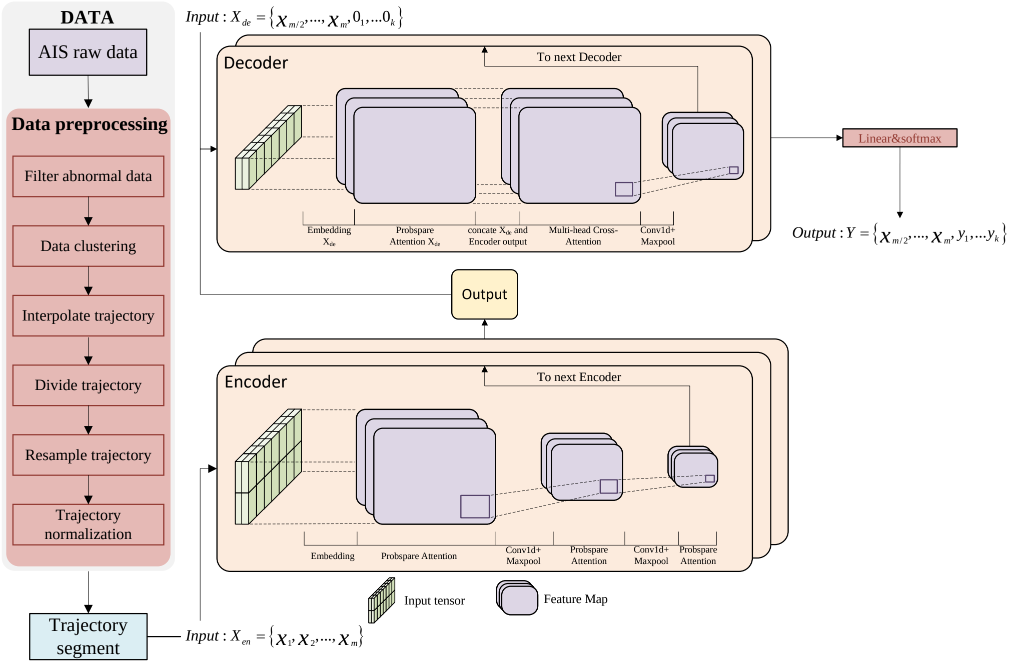


Figure 1. Flowchart of the proposed method.

2.1. Trajectory Prediction

Predicting ship trajectories using AIS data typically involves leveraging a series of historical AIS data to forecast the ship's future trajectory. In this study, we denote all ship time series trajectories as T, which can be represented as follows:

$$\begin{cases} T = \{Traj_1, \dots, Traj_i, \dots, Traj_N\} \\ Traj_i = \left[(t_0, Lat_0, Lon_0, Sog_0, Cog_0), \dots, \left(t_j, Lat_j, Lon_j, Sog_j, Cog_j\right)\right] \end{cases} \tag{1}$$

where $i = 1, \dots, N$, with N being the total number of ship trajectories, and $Traj_i$ representing the i-th trajectory segment. $t_j, Lat_j, Lon_j, Sog_j, Cog_j (j = 1, \dots, n)$ denote the time, latitude, longitude, speed over ground, and course over ground, respectively, of the j-th trajectory point, with n representing the total number of trajectory points in a segment. AIS data comprise numerous trajectory segments, and thus, AIS data containing N segments can be expressed as $T = \{Traj_i\}_{i=1}^{N}$, where $Traj_i$ corresponds to the trajectory segment as defined in Equation (1). In this study, each segment is fed to the model for training, and based on m consecutive trajectory points $\{x_1, x_2, \dots, x_m\}$ in a segment, the model predicts the next k consecutive trajectory points $\{y_{m+1}, y_{m+2}, \dots, y_{m+k}\}$.

2.2. Informer-TP

Informer-TP, based on the Informer model, uses AIS data to predict future ship trajectories. The model extracts historical trajectory features to generate future trajectories. Trajectory prediction is a specific type of time series prediction task. While general time series prediction tasks forecast values within a specific future time frame, trajectory prediction requires forecasting the trajectories of numerous ships at different times. Therefore, we modify the Informer model to handle data structures that predict multiple ship trajectories simultaneously. We divide the trajectory data into independent segments, ignoring interactions between ships to ensure continuity within segments and independence between segments. Each segment's time series is separated and then combined with the corresponding trajectory segment as a whole. All trajectory segments are then fed into the model. The model's computation structure is modified to adapt to this input structure, enabling the simultaneous prediction of different ship trajectories. The structure of the Informer model is illustrated in Figure 2. We can see that it is mainly composed of Encoder and Decoder, where embedding is responsible for encoding the input data to obtain the input representation of the model, and ProbSparse Self-Attention is the unique attention mechanism of Informer. A detailed introduction will be given below.

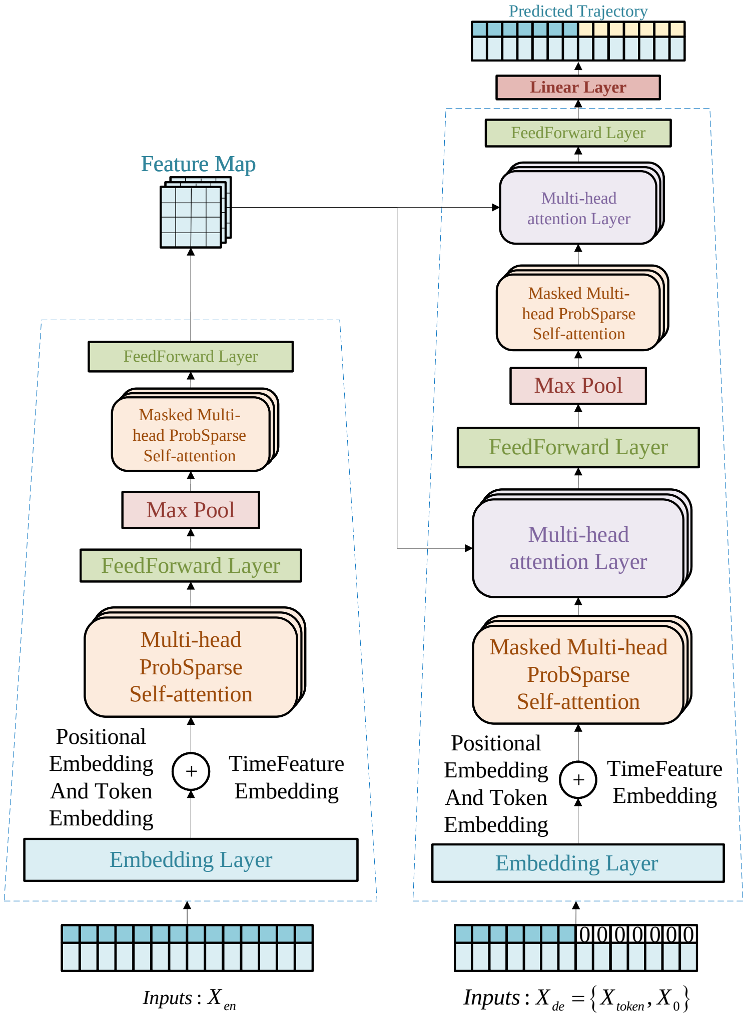

Predicted Trajectory
Linear Layer
FeedForward Layer
Multi-head attention Layer
Masked Multi-head ProbSparse Self-attention
Max Pool
FeedForward Layer
Multi-head attention Layer
Masked Multi-head ProbSparse Self-attention
Positional Embedding And Token Embedding
+
TimeFeature Embedding
Embedding Layer
0 0 0 0 0 0 0
Inputs : $X_{de} = \{X_{token}, X_0\}$
Feature Map
FeedForward Layer
Masked Multi-head ProbSparse Self-attention
Max Pool
FeedForward Layer
Multi-head ProbSparse Self-attention
Positional Embedding And Token Embedding
+
TimeFeature Embedding
Embedding Layer
Inputs : X_{en}

Figure 2. Structure of the Informer model.

2.2.1. Input Representation of Informer

Inputting trajectory segments into Informer, each trajectory segment represents several sequences of trajectories with a length of m, $\{x_1, x_2, \ldots, x_m\}$ where x represents a trajectory point within the trajectory sequence. Initially, scalar projection, local timestamps, and global timestamps are used to transform the input information. The input representation of Informer is illustrated in Figure 3. In this representation, the local timestamp PE embeds contextual information by a fixed position embedding, as shown in Equation (2).

$$\begin{aligned} \mathrm{PE}_{(pos,2j)} &= sin\left(pos/(2L_x)^{2j/d_{model}}\right) \\ \mathrm{PE}_{(pos,2j+1)} &= cos\left(pos/(2L_x)^{2j/d_{model}}\right) \end{aligned} \quad , \tag{2}$$

where pos represents the positional index, $j \in \left\{1, \ldots, \frac{d_{model}}{2}\right\}$ denotes the dimension of embedding, L_x is the length of the trajectory sequence, and d_{model} is the dimension of the model. Subsequently, a learnable global timestamp SE is used to capture the global context. In this study, the time interval between each trajectory point is 5 min; thus, the global timestamp is precise to minute granularity. Five time granularities, including minute, hour, week, month, and year, are used to map the corresponding time features of trajectory points to the interval [−0.5, 0.5]. To align dimensions, a one-dimensional convolutional filter projects scalar contexts to d_{model} dimensional vectors, resulting in u^t_i. Finally, the local timestamp, scalar projection, and global timestamp are combined to form the ultimate input representation. Assuming there are t trajectory sequences X^t and p types of global timestamps SE, the calculation of the input representation vector $\mathcal{X}^t_{\text{feed}[i]}$ for the t-th trajectory sequence is illustrated in Equation (3).

$$\mathcal{X}^t_{\text{feed}[i]} = \alpha u^t_i + \text{PE}_{(L_x \times (t-1)+i,)} + \sum_p [\text{SE}_{(L_x \times (t-1)+i)}]_p \tag{3}$$

where $i \in \{1, \ldots, L_x\}$, α is a factor used to balance scalar projection, local timestamps, and global timestamps, u^t_i represents scalar projection, PE denotes local timestamps, and SE indicates global timestamps. $[]_p$ represents the p-th type of SE. Consequently, the trajectory segment is modified to be represented by scalar projection, local timestamps, and global timestamps.

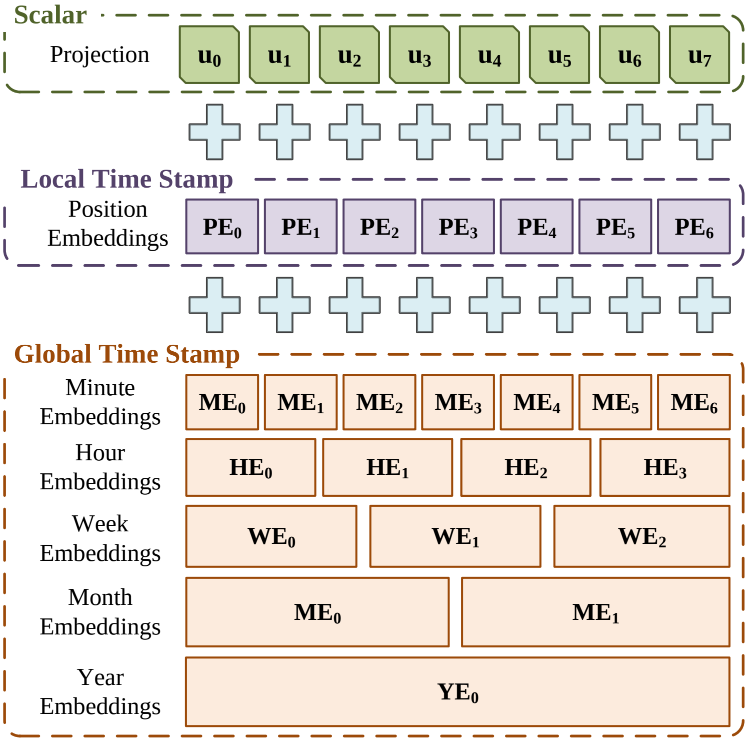


Figure 3. Input Representation of Informer.

2.2.2. ProbSparse Self-Attention

After the trajectory segment undergoes the input representation in Informer, it first enters the ProbSparse Self-Attention layer of the Encoder, where linear transformations are applied to $\mathcal{X}_{\text{feed}}$ to obtain three vectors: **Q**, **K**, and **V**. Here, **Q** represents the query vector, **K** represents the key vector, and **V** represents the value vector. Based on the **Q** vector, a weight distribution is computed by calculating the similarity between the **Q** vector and all **K** vectors, which is then used to perform a weighted sum of associated **V** vectors. The calculation formula is depicted in Equation (4).

$$\begin{cases} \mathbf{Q} = \mathcal{X}_{\text{feed}} W^{\mathbf{Q}} \\ \mathbf{K} = \mathcal{X}_{\text{feed}} W^{\mathbf{K}} \\ \mathbf{V} = \mathcal{X}_{\text{feed}} W^{\mathbf{V}} \end{cases} \tag{4}$$

where $\mathcal{X}_{\text{feed}}$ represents the encoded trajectory sequence, $W^{\mathbf{Q}}$, $W^{\mathbf{K}}$ and $W^{\mathbf{V}}$ are three trainable parameter matrices. Linear transformations are performed by multiplying $\mathcal{X}_{\text{feed}}$ with these matrices individually.

The ProbSparse Self-Attention samples are based on the probability of queries obtained through linear transformation $(\mathbf{Q}, \mathbf{K}, \mathbf{V})$ vectors, enabling the model to focus more on important queries. This prioritizes trajectory points that have a significant impact on subsequent trajectories, assigning them higher weights. To measure the sparsity of queries, a measurement method based on KL (Kullback–Leibler) divergence is introduced to assess the similarity between the attention probability distribution of queries and a uniform distribution. This measurement identifies queries that dominate the self-attention distribution. The sparsity evaluation of the *i*-th query is expressed as shown in Equation (5).

$$M(\mathbf{q}_i, \mathbf{K}) = \ln \sum_{j=1}^{L_K} e^{\frac{\mathbf{q}_i \mathbf{k}_j^\top}{\sqrt{d}}} - \frac{1}{L_K} \sum_{j=1}^{L_K} \frac{\mathbf{q}_i \mathbf{k}_j^\top}{\sqrt{d}} \tag{5}$$

where $\mathbf{q}_i$ is the *i*-th row of $\mathbf{Q}$, $\mathbf{k}_j$ is the *j*-th row of $\mathbf{K}$, L_K is the number of keys, and d is a constant. To simplify the calculation of the sparsity measure $M(\mathbf{q}_i, \mathbf{K})$, an approximation method based on the difference between the maximum and mean values is employed, as shown in Equation (6).

$$\overline{M}(\mathbf{q}_i, \mathbf{K}) = \underset{j}{max} \left\{ \frac{\mathbf{q}_i \mathbf{k}_j^\top}{\sqrt{d}} \right\} - \frac{1}{L_K} \sum_{j=1}^{L_K} \frac{\mathbf{q}_i \mathbf{k}_j^\top}{\sqrt{d}} \tag{6}$$

where max_j represents taking the maximum value of all $\mathbf{k}_j$, and $\sum_j$ represents summing all $\mathbf{k}_j$. By randomly sampling $u = L_\mathbf{K} \ln L_\mathbf{Q}$ dot products, then selecting the top u queries from these sampled dot products, these queries have the highest sparsity measure values $M(\mathbf{q}_i, \mathbf{K})$. Based on this sparsity measure method, ProbSparse Self-Attention attends to the u most significant queries for each key, as shown in Equation (7).

$$\mathcal{A}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}\left(\frac{\overline{\mathbf{Q}}\, \mathbf{K}^\top}{\sqrt{d}} \right) \mathbf{V} \tag{7}$$

where $\overline{\mathbf{Q}}$ is a sparse matrix containing only the top u queries selected based on the sparsity measure $M(\mathbf{q}_i, \mathbf{K})$. For the remaining queries, the average value of the $\mathbf{V}$ vectors is used instead. This enables the model to better focus on trajectory points with significant turns or changes in velocity, while accurately predicting trajectory points with straight-line uniform motion as well.

2.2.3. Encoder

Multiple ProbSparse Self-Attention layers form an attention block, and Informer's Encoder consists of multiple attention blocks and distillation layers. After encoding, the trajectory sequence is inputted into the attention block to calculate the $(\mathbf{Q}, \mathbf{K}, \mathbf{V})$ vectors. Then, dimension reduction is achieved through max-pooling and one-dimensional convolution to halve the input dimensions, thus implementing the Distilling operation. The Distilling operation from layer j to layer $j + 1$ is illustrated in Equation (8).

$$\mathbf{X}_{j+1}^t = \text{MaxPool}\Big(\text{ELU}\Big(\text{Conv1d}\Big(\Big[\mathbf{X}_j^t \Big]_{AB} \Big) \Big) \Big) \tag{8}$$

where $\mathbf{X}_j^t$ represents the input feature map of the *j*-th layer, $[]_{\text{AB}}$ denotes the attention block, Conv1d is a one-dimensional convolution operation, ELU is the exponential linear unit activation function, and MaxPool represents the max-pooling operation. After multiple Distilling operations in the Encoder, the feature map of the trajectory sequence is obtained. It stores important features of past data in the trajectory sequence. By stacking multiple Encoders, reducing input length and decreasing attention blocks between each Encoder, and then concatenating the feature maps of all Encoders, the final output of the Encoder is obtained, completing the encoding operation on the trajectory sequence.

2.2.4. Decoder

The Decoder of Informer consists of multiple DecoderLayers stacked with LayerNorm and Linear Layer. Each DecoderLayer is composed of a stack of Multi-head Attention and Masked Multi-head ProbSparse Self-Attention. The input of the Decoder consists of two parts: the first input is the output of the Encoder, and the second input is a trajectory sequence, where half of the Encoder's output is concatenated with another trajectory sequence of the same shape as the prediction target, but with values set to 0. The input of the Decoder is represented as shown in Equation (9).

$$\mathbf{X}_{\text{de}}^{t} = \text{Concat}\left(\mathbf{X}_{\text{token}}^{t}, \mathbf{X}_{0}^{t}\right) \in \mathbb{R}^{\left(L_{\text{token}}+L_{y}\right)\times d_{\text{model}}} \tag{9}$$

where $\mathbf{X}_{\text{token}}^{t} \in \mathbb{R}^{(L_{token} \times d_{model})}$ is the start token sequence, $\mathbf{X}_{0}^{t} \in \mathbb{R}^{L_{y} \times d_{model}}$ is the sequence to be predicted, which is set to 0, L_{token} is the size of the start token sequence, and L_{y} is the size of the predicted sequence. The length of the Decoder's output is consistent with the length of the second input part. The entire model aims to predict the masked part of the Decoder's output. The output of the first DecoderLayer serves as the query, while the output of the Encoder serves as the key and value. Through subsequent DecoderLayers, vectors for each position to be predicted are obtained. After decoding through layers of the Decoder, the true trajectory sequence $\{x_1, x_2, \ldots, x_m\}$ is concatenated with the predicted trajectory sequence $\{y_{m+1}, y_{m+2}, \ldots, y_{m+k}\}$ and outputted together. Here, k represents the number of predicted trajectory points. This enables the model to directly predict all outputs through a forward process, avoiding accumulation errors in stepwise inference. Additionally, the input between each DecoderLayer is halved, increasing prediction efficiency.

2.2.5. Loss Function

The error between the actual sequence y_t and the predicted sequence y_p is calculated using the loss function (10). By minimizing the loss value, weight parameters are adjusted to improve the prediction accuracy. The loss function is expressed as follows:

$$Loss = \frac{\sum_{i=1}^{n}\left(y_{i}^{t} - y_{i}^{p}\right)^{2}}{n} \tag{10}$$

where n represents the length of the training set, y_i^t denotes the true value, and y_i^p denotes the predicted value.

3. Experiment and Results

3.1. Data Processing

In this study, data from coastal areas were selected for analysis. The data were sourced from the publicly available vessel trajectory data website, U.S. Coast Guard (https://marinecadastre.gov/ais/) (accessed on 27 July 2024), and covered the period from 1 June 2022 to 31 August 2022. The latitude and longitude ranges were (84°–77° W) and (23°–27° N), respectively. AIS data of cargo ships with a normal navigation status were used for training, validation, and testing. Table 1 shows the raw AIS data, and Table 2 provides the data description.

Table 1. Raw AIS data.

MMSI	BaseDateTime	LAT (°)	LON (°)	SOG (kn)	COG (°)	VesselType	Status
377908000	2022-06-01T00:00:06	25.64317	−80.02941	11.1	181.3	70.0	0.0
377908000	2022-06-01T00:02:37	25.63539	−80.02976	11.2	184.6	70.0	0.0

Table 2. Description of AIS Data.

Name	Description
MMSI	Maritime Mobile Service Identity value
BaseDataTime	Full UTC date and time
LAT	Latitude
LON	Longitude
SOG	Speed over ground
COG	Course over ground
VesselType	Vessel type
Status	Navigation status

Due to various physical factors such as signal loss and equipment failure, AIS data often suffer from issues like data loss and duplication. Additionally, adverse weather conditions and rough seas can significantly disrupt ship navigation, leading to abnormal AIS data. Therefore, it is essential to clean the AIS data. To address these abnormalities, this paper presents a comprehensive data processing method to effectively improve data quality:

(1) The ship is considered stationary if the LON and LAT of two or more consecutive trajectory points are equal or the SOG is less than 1. At this time, the AIS data are duplicate data or abnormal data with minor changes. Therefore, it is necessary to remove duplicate data and set thresholds for SOG and COG to eliminate abnormal data. The minimum SOG threshold is 1. Since the maximum speed of cargo ships generally does not exceed 30, the maximum value is set to 30. The COG threshold ranges from 0 to 360. Data outside these ranges are removed. Finally, trajectories are separated based on MMSI to obtain the independent trajectories for each ship.

(2) For each ship's trajectory, if the time interval between two consecutive points is less than 60 min, it is considered a single trajectory segment; otherwise, it is divided into new segments. In our study, we only observe ships with continuous sailing times longer than 4 h; therefore, we remove trajectory segments with fewer than 240 data points or a duration of less than 4 h.

(3) Detect abnormal ship trajectories using DBSCAN. First, use the K-distance algorithm to determine suitable parameters for DBSCAN clustering, then apply DBSCAN to achieve optimal clustering. Remove outlier segments identified in the clustering results.

(4) Cubic spline interpolation generates smooth curves between discrete data points. Since the ship's LON and LAT change frequently, we use cubic spline interpolation between consecutive points in each trajectory segment to interpolate missing LAT and LON values and maintain time intervals of less than 1 min. For SOG and COG that change relatively smoothly, we average the SOG and COG from adjacent points. Segments with an average SOG of less than 2 are considered abnormal and removed.

(5) Resample trajectory points at 5 min intervals by averaging multiple AIS data points within each interval to maintain consistent spacing. This interval can be adjusted based on the desired precision of the final results to capture finer trajectory details. Smaller time intervals result in higher trajectory accuracy.

(6) Use the min–max normalization method for normalization. The normalization formula is as follows:

$$X_{\text{norm}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{11}$$

where X is the original data, $X_{\min}$ is the minimum value of the original data, $X_{\max}$ is the maximum value of the original data, and X_{norm} is the normalized data, ranging between [0, 1].

(7) For the Informer-TP method, each trajectory segment is divided into new segments based on the sum of historical trajectory length, L_{token} length, and prediction length, with time features separately extracted and input into the Informer-TP along with the trajectory segments. Other models are trained by dividing trajectory segments according to input

and output lengths, where the input length corresponds to L_{token} length in Informer-TP and the output length is the prediction length.

After preprocessing, the total number of ships is 1108, the total number of ship trajectories is 3898, and the total number of trajectory points is 441,125. The dataset is divided into training, validation, and test sets in an 8:1:1 ratio. The training set contains 882 ships and 3118 trajectories with a total of 300,143 trajectory points, the validation set contains 119 ships and 390 trajectories with a total of 65,662 trajectory points, and the test set contains 107 ships and 390 trajectories with a total of 75,320 trajectory points.

3.2. Evaluation Metrics

The Haversine (HAV) distance is a spherical geometry-based method for calculating the shortest distance between two points on the Earth's surface. In ship trajectory prediction tasks, it can calculate the actual distance between two latitude and longitude points, providing an intuitive evaluation of prediction accuracy. Therefore, this paper primarily uses HAV as the metric for assessing model prediction accuracy. The Mean Squared Error (MSE) measures the average squared difference between the predicted and actual trajectory points, calculating the average of the squares of these differences. The Mean Absolute Error (MAE) measures the average distance between predicted and actual trajectory points, reflecting the average error size. These three metrics—HAV, MSE, and MAE—offer a comprehensive, accurate, and intuitive assessment of model performance, aiding in model improvement and optimization. Lower values indicate higher model accuracy. The formulas are shown in Equations (12)–(14).

$$HAV = 2Rarcsin\left(\sqrt{sin^2(\bar{\phi}) + cos(\phi_1)cos(\phi_2)sin^2(\bar{\lambda})}\right) \tag{12}$$

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

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{14}$$

In these formulas, R is the Earth's radius, $\bar{\phi} = |\phi_2 - \phi_1|/2, \bar{\lambda} = |\lambda_2 - \lambda_1|/2$, where ϕ_2 is the actual longitude, ϕ_1 is the predicted longitude, λ_2 is the actual latitude, and λ_1 is the predicted latitude. n denotes the number of samples, y_i is the actual value of the i-th sample, and $\hat{y}_i$ is the predicted value of the i-th sample.

3.3. Experimental Environment and Parameter Settings

This study was conducted on a Windows 10 operating system using Pycharm as the programming software and Torch 2.1.1 as the deep learning framework, with Python 3.9. All experiments were performed on a 3.5GHz AMD Ryzen 5 5600 CPU and an NVIDIA RTX 4060 GPU (16GB RAM) (Made in the Santa Clara, CA, USA by AMD and NVIDIA).

In Informer-TP, the Encoder and Decoder modules have 2 and 1 layers, respectively. The number of attention heads is set to 8, the model size is 512, the fully connected layer size is 2048, and the Probsparse attention coefficient is 3. The model uses Mean Squared Error (MSE) as the loss function, with the Adam optimizer. The initial learning rate is set to 5×10^{-6} and will adaptively decay during training. An early stopping mechanism halts training if the MSE does not decrease for 5 consecutive epochs. The number of epochs is set to 100, and the batch size is 32.

For the baseline models, the Transformer model's parameters largely align with those of the Informer-TP model. For other baseline models, the maximum number of epochs is set to 200, the batch size to 240, and the early stopping mechanism is consistent with that of Informer-TP. We found optimal performance for the LSTM layer with a hidden size of 64 and a 2-layer Decoder and Encoder. The ConvLSTM-Seq2Seq model performed best with a kernel size of 3, and the LSTM-Seq2Seq-Attention model with 4 attention layers.

After determining the optimal parameters for these three baseline models, experiments with different activation functions revealed that LSTM-Seq2Seq, LSTM-Seq2Seq-Attention, and ConvLSTM-Seq2Seq models performed best with Tanh, SoftPlus, and ELU activation functions, respectively. All models are configured with these optimal parameters, as detailed in Tabel 3.

Table 3. Experimental parameter settings for the 5 models.

Model	Parameter Name	Parameter Size
LSTM-Seq2Seq	Hidden size	64
	Layer number	2
	Activation function	Tanh
LSTM-Seq2Seq-Attention	Hidden size	64
	Layer number	2
	Activation function	Softplus
	Attention layers	4
ConvLSTM-Seq2Seq	Hidden size	64
	Layer number	2
	Activation function	ELU
	Kernel size	(1,2,3)
Transformer	Model dimension	512
	Head number	8
	Activation function	Gelu
	Attention type	Self-attention
Informer-TP	Model dimension	512
	Head number	8
	Activation function	Gelu
	Attention type	ProbSparse-Attention

3.4. Results

3.4.1. Model Comparison

To verify the effectiveness of Informer-TP, we compare the proposed methods with LSTM-Seq2Seq, LSTM-Seq2Seq-Attention, ConvLSTM-Seq2Seq, and Transformer models. The dataset processing is consistent with the description in Section 3.1, and the experimental parameters match those in Section 3.3. For each method, the input lengths are fixed at 36, 42, and 24, while the output lengths are set to 24, 18, and 12, respectively.

(1) LSTM-Seq2Seq [20]: This model uses LSTM units to capture long-term dependencies in sequence data, enabling it to leverage past information for future predictions. Its recursive mechanism enhances the understanding of the structure and patterns within the sequence data.

(2) LSTM-Seq2Seq-Attention [21]: The addition of the attention mechanism improves the LSTM-Seq2Seq model by dynamically focusing on relevant parts of the input sequence during output generation, enhancing both encoding and decoding capabilities.

(3) ConvLSTM-Seq2Seq [19]: This model combines convolution operations to extract spatial features with LSTM units to retain long-term dependencies. It effectively captures local structures and patterns, providing strong representation and prediction capabilities for time series data.

(4) Transformer [36]: Utilizing a self-attention mechanism, the Transformer model dynamically focuses on different parts of the input sequence while generating the output. It effectively captures long-range dependencies without being constrained by sequence length, using positional encoding to maintain the order of positions in the sequence.

Table 4 shows the prediction results of the five models under different input-output conditions. Informer-TP consistently performs best across all settings, followed by the Transformer model. For predicting 12 points, the errors of the Transformer and Informer-TP models are very close, whereas ConvLSTM-Seq2Seq performs the worst. As prediction length increases, errors for all models increase, but Informer-TP shows a smaller error increase compared to the Transformer, indicating better long-term prediction effectiveness. ConvLSTM-Seq2Seq has the smallest overall error increase, but it shows the largest error increase when predicting 18 points and the smallest when predicting 24 points. Similarly, LSTM-Seq2Seq-Attention has a smaller error increase when predicting 24 points than when predicting 18 points. This may be because more input points are required for predicting 18 points than for 24 points, and these models are more sensitive to longer input sequences, leading to greater error due to interference. Despite this, the overall error increase for ConvLSTM-Seq2Seq and LSTM-Seq2Seq-Attention is still smaller than that for LSTM-Seq2Seq, making them more effective for longer predictions. ConvLSTM-Seq2Seq, in particular, shows a smaller overall error increase compared to LSTM-Seq2Seq-Attention, making it better for longer sequence predictions.

Table 4. Prediction results of the 5 models on the test set for different input and output points.

Number of Forecast Points	Model	MSE (10^{-4})	MAE (10^{-2})	HAV (km)
$p = 24, q = 12$	LSTM-Seq2Seq	1.2023	0.6024	0.9948
	LSTM-Seq2Seq-Attention	2.0374	0.7101	1.1768
	ConvLSTM-Seq2Seq	3.1505	0.8427	1.3890
	Transformer	**0.8154**	0.4201	0.7332
	Informer-TP	0.8272	**0.4176**	**0.7288**
$p = 42, q = 18$	LSTM-Seq2Seq	2.7829	0.8612	1.4204
	LSTM-Seq2Seq-Attention	4.2680	1.0219	1.6850
	ConvLSTM-Seq2Seq	7.0358	1.1840	1.9660
	Transformer	2.7212	0.7467	1.2560
	Informer-TP	**2.5064**	**0.7008**	**1.1883**
$p = 36, q = 24$	LSTM-Seq2Seq	8.3109	1.6781	1.9069
	LSTM-Seq2Seq-Attention	10.2782	1.7757	1.9892
	ConvLSTM-Seq2Seq	7.2701	1.3528	2.0011
	Transformer	6.2514	1.1352	1.8776
	Informer-TP	**5.4873**	**1.0711**	**1.7829**

To further evaluate the performance of different models in predicting trajectories of varying lengths in different scenarios, this paper selects linear and turning trajectories in four different scenarios to predict new trajectories and records the maximum, minimum, and average values of the three metrics in each scenario. The prediction results are shown in Tables 5–7, and the visualization results are shown in Figures 4–6.

From Tables 5–7, it can be observed that when predicting 12 points, the Transformer performs better than the Informer-TP. However, as the prediction length increases, Informer-TP's performance surpasses all other models. When using 42 input points to predict 18 points, the errors for the three Seq2Seq models increase significantly. Conversely, when using 36 input points to predict 24 points, the prediction errors decrease, sometimes even outperforming the Transformer. This indicates that these three models are highly sensitive to the input length. Overall, Informer-TP demonstrates the best performance and has a distinct advantage in long-sequence predictions.

Table 5. Prediction results of the 5 models for the next 12 points.

Model	Statistics	MSE (10^{-4})	MAE (10^{-2})	HAV (km)
LSTM-Seq2Seq	Max	0.6344	0.6483	1.2278
	Min	0.0705	0.2312	0.3709
	Average	0.3279	0.4710	0.7901

Table 5. *Cont.*

Model	Statistics	MSE (10^{-4})	MAE (10^{-2})	HAV (km)
LSTM-Seq2Seq-Attention	Max	0.7646	0.7742	1.3353
	Min	0.0760	0.2230	0.3786
	Average	0.4489	0.5444	0.9202
ConvLSTM-Seq2Seq	Max	2.6075	1.4190	2.2840
	Min	0.0507	0.1762	0.2945
	Average	0.8587	0.6470	1.0595
Transformer	Max	**0.3178**	**0.4301**	**0.7383**
	Min	**0.0099**	**0.0695**	**0.1843**
	Average	**0.0934**	**0.1777**	**0.3460**
Informer-TP	Max	0.3360	0.4522	0.7584
	Min	0.0141	0.0862	0.1973
	Average	0.1213	0.2306	0.4230

Table 6. Prediction results of the 5 models for the next 18 points.

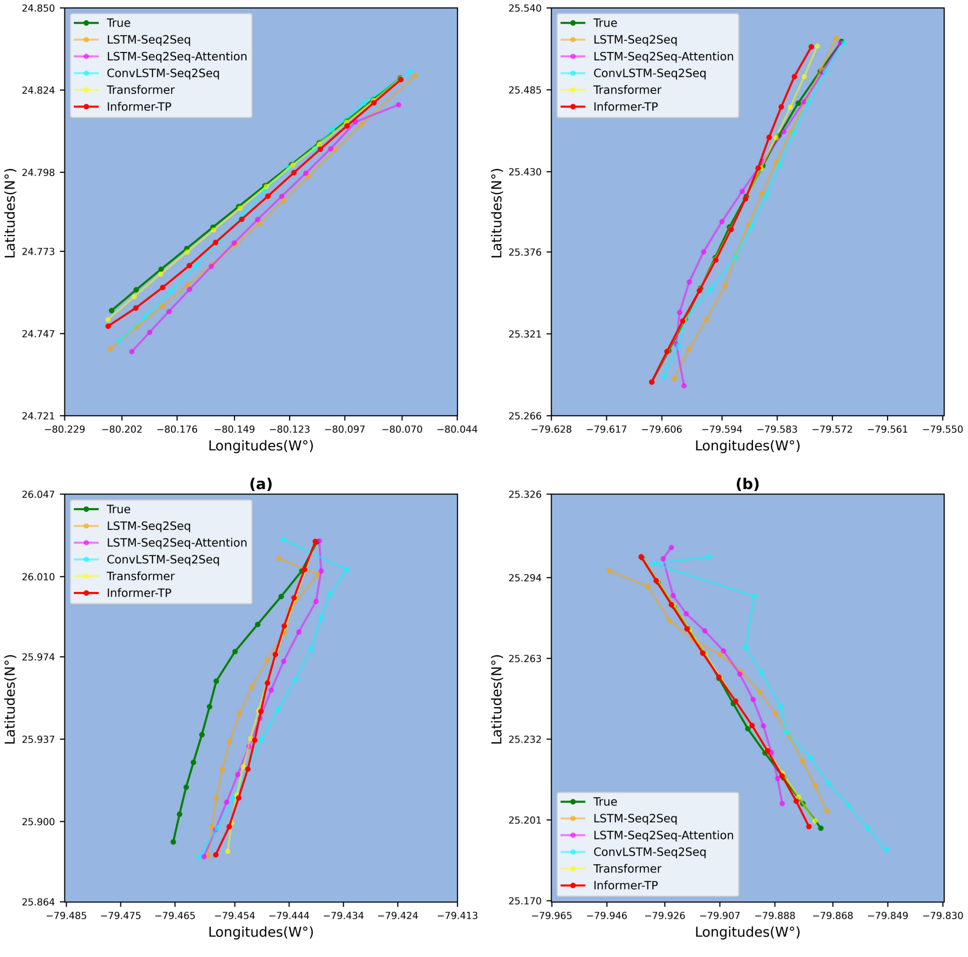
Model	Statistics	MSE (10^{-4})	MAE (10^{-2})	HAV (km)
LSTM-Seq2Seq	Max	5.2781	2.1135	3.2104
	Min	0.1151	0.2688	0.4594
	Average	1.4683	0.8132	1.2835
LSTM-Seq2Seq-Attention	Max	4.9573	1.8517	2.9739
	Min	0.1455	0.3423	0.5690
	Average	1.6515	0.8801	1.4802
ConvLSTM-Seq2Seq	Max	4.0741	1.3873	2.5666
	Min	0.0586	0.2048	0.3201
	Average	1.1498	0.5992	1.0659
Transformer	Max	0.6558	0.5102	0.9139
	Min	0.1267	0.3025	0.5245
	Average	0.3738	0.4239	0.7361
Informer-TP	Max	**0.1626**	**0.3043**	**0.5163**
	Min	**0.0331**	**0.1091**	**0.2538**
	Average	**0.1183**	**0.2372**	**0.4252**

Table 7. Prediction results of the 5 models for the next 24 points.

Model	Statistics	MSE (10^{-4})	MAE (10^{-2})	HAV (km)
LSTM-Seq2Seq	Max	3.4489	1.3835	2.4952
	Min	**0.1239**	**0.2687**	**0.4542**
	Average	1.0370	0.6073	1.0589
LSTM-Seq2Seq-Attention	Max	**0.7626**	0.6816	1.1077
	Min	0.2097	0.3857	0.6449
	Average	**0.5092**	0.5773	0.9424
ConvLSTM-Seq2Seq	Max	2.7302	1.3659	2.1866
	Min	0.1444	0.2720	0.5292
	Average	1.1275	0.7266	1.2244
Transformer	Max	1.8381	1.0266	1.5882
	Min	0.8385	0.5860	1.0264
	Average	1.1311	0.7809	1.2748

Table 7. *Cont.*

Model	Statistics	MSE (10^{-4})	MAE (10^{-2})	HAV (km)
Informer-TP	Max	0.9524	**0.6545**	**1.0622**
	Min	0.1907	0.2845	0.5275
	Average	0.5151	**0.4649**	**0.7919**



(a) (b) (c) (d)

Figure 4. Prediction results of the 5 models for 12 points in different scenarios. (**a**) Predicted trajectory of the next 12 points for a ship navigating towards the lower left; (**b**) Predicted trajectory of the next 12 points for a ship navigating towards the upper right; (**c**) Predicted trajectory of the next 12 points for a ship curving towards the lower left; (**d**) Predicted trajectory of the next 12 points for a ship navigating towards the lower right.

In Figure 4a, all models demonstrate relatively stable performance, with Informer-TP, Transformer, and ConvLSTM-Seq2Seq showing good fitting to the real trajectories. In Figure 4b, Informer-TP and Transformer effectively fit the early trajectories, while the other three models perform better in fitting the later trajectories. In Figure 4c, all models exhibit some degree of deviation, with LSTM-Seq2Seq and ConvLSTM-Seq2Seq showing larger initial position biases. In Figure 4d, all three Seq2Seq models show unstable trajectories, particularly ConvLSTM-Seq2Seq, which exhibits more fluctuation in trajectory trends. Overall, only Informer-TP and Transformer maintain stable trajectories across various scenarios, while the other three models fail to fit all scenarios well.

In Figure 5a, only the trajectory length of Informer-TP closely matches the length of the real trajectory, while trajectories of other models are longer than the real trajectory. In Figure 5b, all models can fit the real trajectory well. In Figure 5c, none of the models capture the initial turning trend, and only Informer-TP shows a trajectory length more like the real one. In Figure 5d, trajectories of Informer-TP and Transformer remain stable, while trajectories of other models exhibit significant deviations, with LSTM-Seq2Seq-Attention showing the largest deviation. Overall, Informer-TP demonstrates increasing advantages; even as the prediction length increases, it can still fit the real trajectory well.

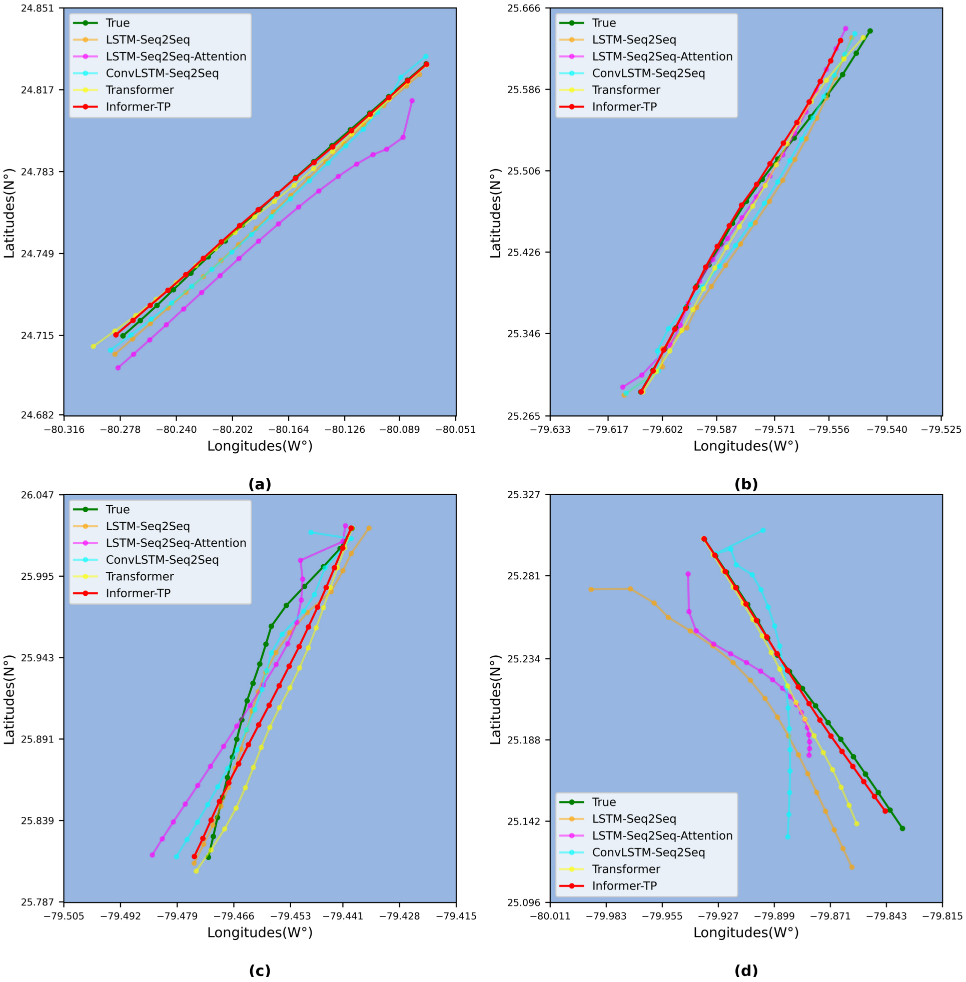
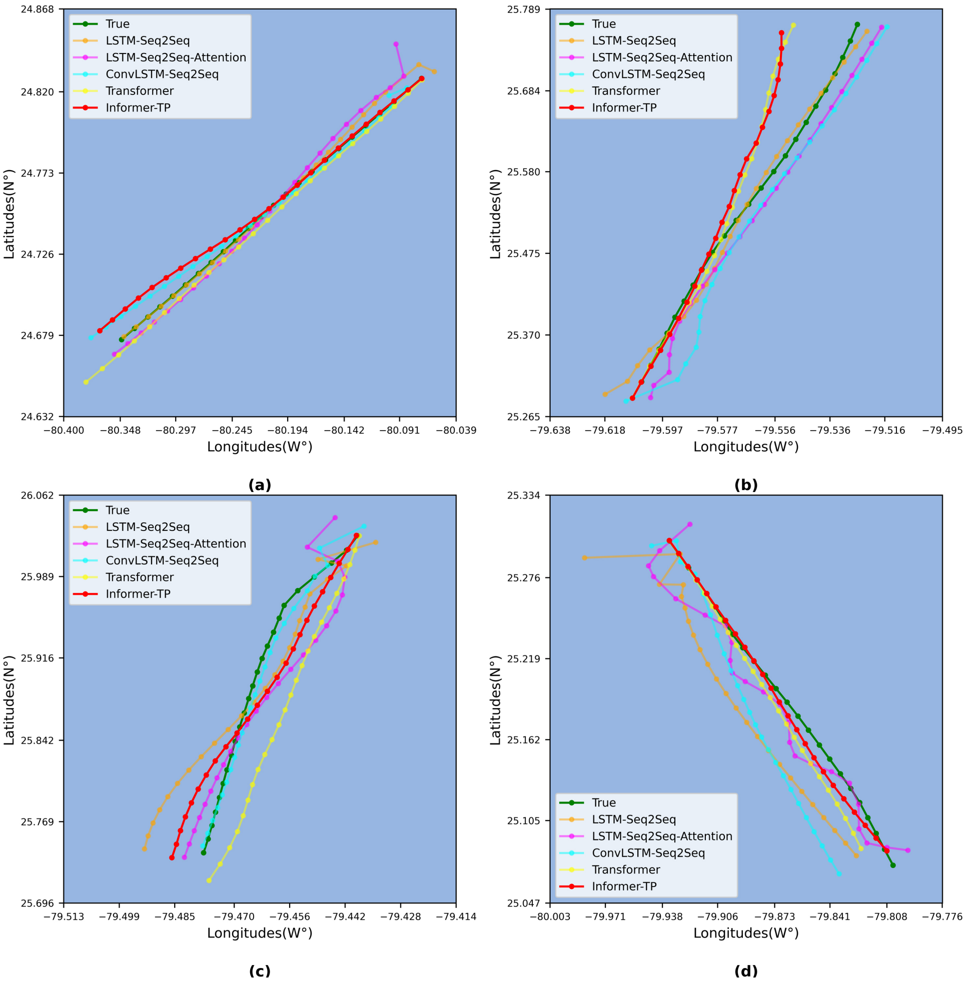


Figure 5. Prediction results of the 5 models for 18 points in different scenarios. (**a**) Predicted trajectory of the next 18 points for a ship navigating towards the lower left; (**b**) Predicted trajectory of the next 18 points for a ship navigating towards the upper right; (**c**) Predicted trajectory of the next 18 points for a ship curving towards the lower left; (**d**) Predicted trajectory of the next 18 points for a ship navigating towards the lower right.

In Figure 6a, Informer-TP exhibits deviation in the latter half, while LSTM-Seq2Seq shows the best fitting, almost overlapping with the real trajectory. In Figure 6b, Informer-TP also shows a deviation in the latter half, while LSTM-Seq2Seq overall fits better. In Figure 6c, although Informer-TP captures the trend, there is still some deviation overall, with ConvLSTM-Seq2Seq showing the best fitting. In Figure 6d, only Informer-TP fits the real trajectory well, while all other models exhibit deviations. In summary, when predicting long sequences, all models perform well in some scenarios, but only Informer-TP consistently maintains stable trajectories across all scenarios. Other models tend to exhibit some degree of deviation, indicating that Informer-TP still outperforms them.



(a) (b) (c) (d)

Figure 6. Prediction results of the 5 models for 24 points in different scenarios. (**a**) Predicted trajectory of the next 24 points for a ship navigating towards the lower left; (**b**) Predicted trajectory of the next 24 points for a ship navigating towards the upper right; (**c**) Predicted trajectory of the next 24 points for a ship curving towards the lower left; (**d**) Predicted trajectory of the next 24 points for a ship navigating towards the lower right.

3.4.2. Trajectory Prediction with Different Input and Output Lengths

To evaluate the influence of the L_{token} and output length on the performance of the Informer-TP model, we fix the output length at q = 12, 18, and 24, and set the L_{token} length to p = 18, 24, 30, 36, 42, and 48, respectively. Each output length is tested with four different L_{token} lengths to assess their impact on the model. Other parameters of the model remain consistent apart from the L_{token} and output length.

To assess the overall performance, we validate different L_{token} and output lengths on the test set, recording the average evaluation metrics for each L_{token} and output length combination. The average prediction results for different output points with varying L_{token} sizes on the test set are presented in Table 8.

In Table 8, for an output length of q = 12, the model with L_{token} length p = 24 exhibits the lowest average MAE, and HAV values. For an output length of q = 18, the model with L_{token} length p = 42, and for an output length of q = 24, the model with L_{token} length p = 36 demonstrates the smallest average MSE, MAE, and HAV values. When the output length q = 12, p = 36 has the largest MSE, MAE, and HAV, where the HAV of 0.7935 is 8.87% higher than the HAV of 0.7288 for p=24. When the output length q = 18, p = 24 has the largest MSE, MAE, and HAV, where the HAV of 1.4211 is 19.59% higher than the HAV of 1.1883 for p = 42. When the output length q = 24, p = 30 has the largest HAV and MAE, where the HAV of 2.0678 is 15.98% higher than the HAV of 1.7829 when

p = 36. It can be observed that the performance of the model is influenced by different L_{token} lengths. Among the different L_{token} lengths for the three output lengths, we compare the best-performing results. For q = 12, compared to q = 18, HAV increases by 63.05%, and for q = 24, it increases by 144.63%.

Table 8. Average prediction results for different output points with varying L_{token} sizes on the test set.

Number of Forecast Points	Number of Start Token Points	MSE (10^{-4})	MAE (10^{-2})	HAV (km)
q = 12	p = 36	0.9084	0.4557	0.7935
	p = 30	0.8605	0.4421	0.7680
	p = 24	0.8272	**0.4176**	**0.7288**
	p = 18	**0.8206**	0.4422	0.7686
q = 18	p = 42	**2.5064**	**0.7008**	**1.1883**
	p = 36	2.6451	0.7450	1.2666
	p = 30	2.5647	0.7368	1.2575
	p = 24	2.9189	0.8371	1.4211
q = 24	p = 48	5.8946	1.0983	1.8320
	p = 42	6.4256	1.2132	2.0267
	p = 36	**5.4873**	**1.0711**	**1.7829**
	p = 30	6.3803	1.2270	2.0678

To further investigate the impact of the L_{token} and output length of the model, we conducted experiments on trajectories from four different scenarios. We recorded the maximum, minimum, and average values of each evaluation metric for each experiment group. The prediction results for different L_{token} sizes and output sizes are shown in Tables 9–11, and the predicted results are illustrated in Figures 7–9.

From Tables 9–11, it can be observed that across different scenarios, p = 24 for predicting 12 points, p = 42 for predicting 18 points, and p=36 for predicting 24 points have the smallest maximum and average values among the three metrics. On the other hand, p = 30 for predicting 12, 18, and 24 points has the smallest minimum values among the three metrics. In summary, the model performs best when L_{token} is set to 24, 42, and 36 for predicting 12, 18, and 24 points, respectively.

Table 9. Prediction results for 12 points in different scenarios with varying L_{token} sizes.

Number of Start Token Points	Statistics	MSE (10^{-4})	MAE (10^{-2})	HAV (km)
p = 36	Max	0.5891	0.5784	0.9976
	Min	0.0641	0.2063	0.3583
	Average	0.2665	0.3520	0.6258
p = 30	Max	0.3998	0.5167	0.8418
	Min	**0.0130**	**0.0887**	**0.1998**
	Average	0.1832	0.2824	0.4926
p = 24	Max	**0.3422**	**0.4691**	**0.7745**
	Min	0.0269	0.1448	0.2633
	Average	**0.1542**	**0.2676**	**0.4871**
p = 18	Max	0.4051	0.5239	0.8589
	Min	0.0747	0.2060	0.3674
	Average	0.2067	0.3336	0.5856

Table 10. Prediction Results for 18 Points in Different Scenarios with Varying L_{token} Sizes.

Number of Start Token Points	Statistics	MSE (10^{-4})	MAE (10^{-2})	HAV (km)
$p = 42$	Max	**0.7440**	**0.6309**	**1.0927**
	Min	0.0461	0.1598	0.3144
	Average	**0.3651**	**0.3871**	**0.6669**
$p = 36$	Max	0.8680	0.7205	1.1763
	Min	0.0759	0.2072	0.3757
	Average	0.3737	0.4168	0.7127
$p = 30$	Max	1.0501	0.8249	1.3120
	Min	**0.0266**	**0.1311**	**0.2637**
	Average	0.4559	0.4501	0.7719
$p = 24$	Max	1.7421	0.9906	1.5976
	Min	0.0529	0.1770	0.3147
	Average	0.7285	0.5439	0.8942

Table 11. Prediction results for 24 points in different scenarios with varying L_{token} sizes.

Number of Start Token Points	Statistics	MSE (10^{-4})	MAE (10^{-2})	HAV (km)
$p = 48$	Max	1.8091	0.9716	1.6747
	Min	0.0919	0.1935	0.3501
	Average	0.6866	0.5152	0.8934
$p = 42$	Max	3.4964	1.3182	2.0660
	Min	0.2065	0.3371	0.5483
	Average	1.3686	0.7584	1.2297
$p = 36$	Max	**1.1638**	**0.7365**	**1.3757**
	Min	0.0754	0.2247	0.3887
	Average	**0.4720**	**0.4298**	**0.7752**
$p = 30$	Max	4.7201	1.6097	2.5657
	Min	**0.0440**	**0.1619**	**0.3065**
	Average	1.7028	0.7801	1.2526

In Figure 7a, we can observe that all predicted trajectories exhibit some degree of deviation from the real trajectory. In Figure 7b, the trajectory with $p = 24$ is closest to the real trajectory, while in Figure 7c, $p = 30$ is closest to the real trajectory. In Figure 7d, only $p = 24$ is closest to the real trajectory. When there is a significant turning trend, trajectories predicted with $p = 36$ and $p = 30$ exhibit larger deviations. Although $p = 18$ has slightly less deviation, there is still some distance from the real trajectory. Overall, trajectories predicted with $p = 24$ are the most similar to the real trajectory.

In Figure 8a, trajectories with $p = 42$ and $p = 30$ are relatively close to the real trajectory. In Figure 8b, $p = 24$ aligns more closely with the real trajectory. In Figure 8c, $p = 24$ matches the front portion of the real trajectory but fails to capture the turning trend. In Figure 8d, only $p = 42$ closely resembles the real trajectory, with a similar trajectory length. It can be observed that the trajectory lengths of all trajectories are relatively consistent, but there is some deviation in capturing the turning trends. Overall, trajectories predicted with $p = 42$ are the most similar to the real trajectory.

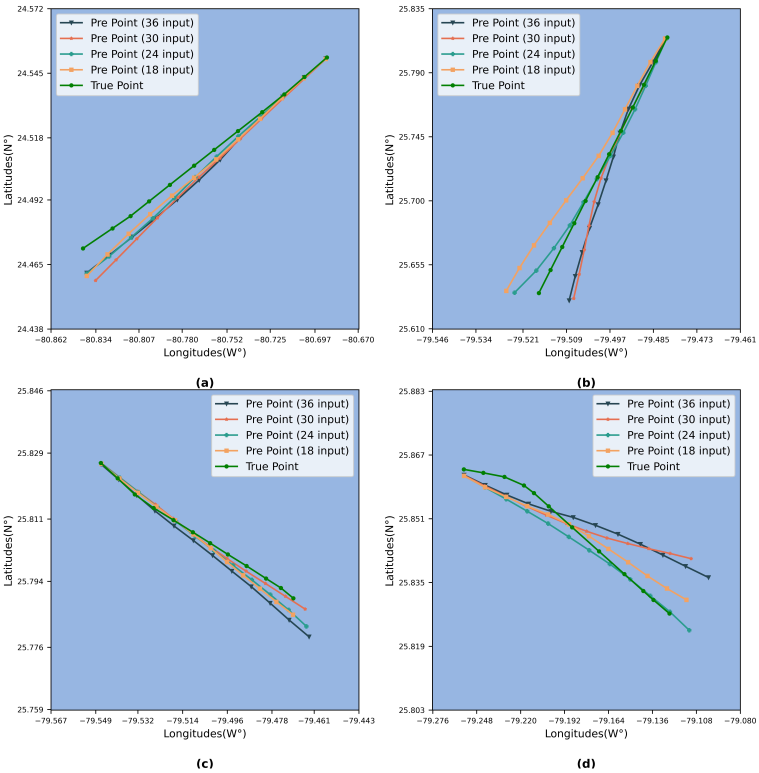


Figure 7. Results of predicting 12 points in different scenarios with varying L_{token} sizes. (**a**,**b**) Predicted trajectory of the next 12 points for a ship navigating towards the lower left; (**c**,**d**) Predicted trajectory of the next 12 points for a ship curving towards the lower right.

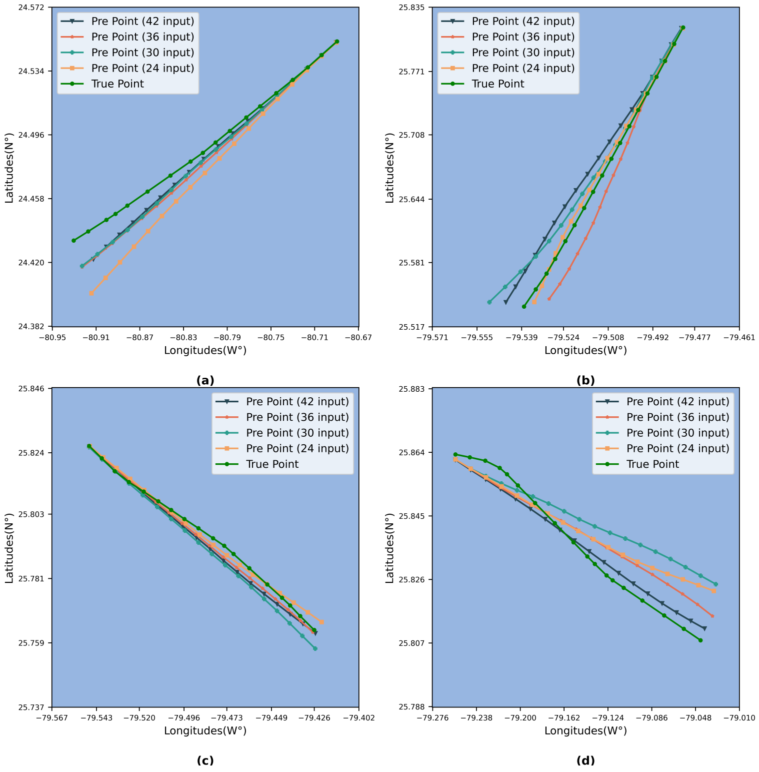
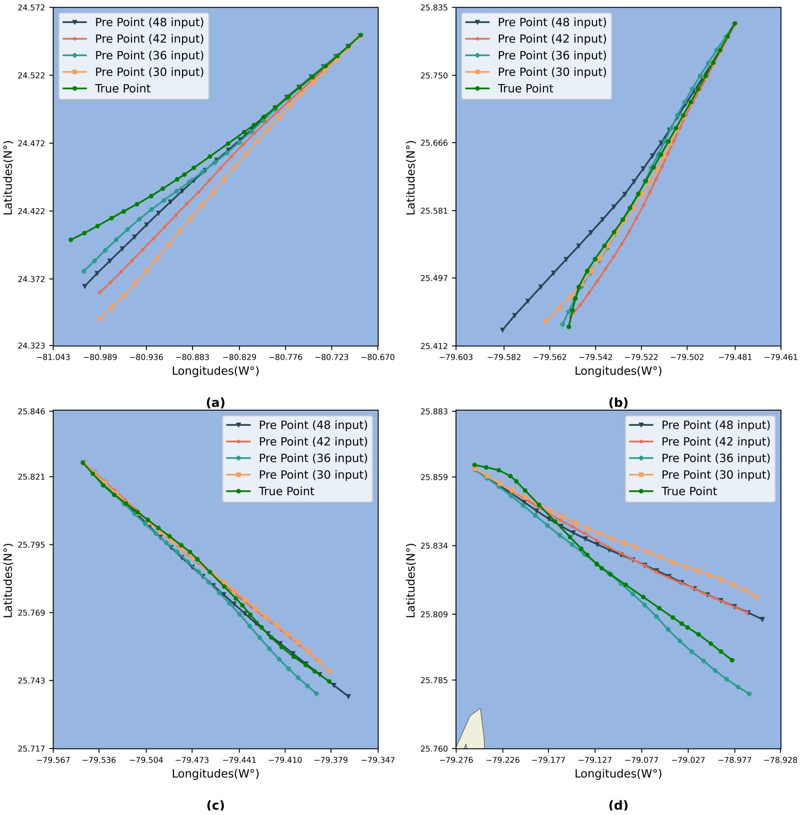


Figure 8. Results of predicting 18 points in different scenarios with varying L_{token} sizes. (**a**,**b**) Predicted trajectory of the next 18 points for a ship navigating towards the lower left; (**c**,**d**) Predicted trajectory of the next 18 points for a ship curving towards the lower right.



(a) **(b)** **(c)** **(d)**

Figure 9. Results of predicting 24 points in different scenarios with varying L_{token} sizes. (**a**,**b**) Predicted trajectory of the next 24 points for a ship navigating towards the lower left; (**c**,**d**) Predicted trajectory of the next 24 points for a ship curving towards the lower right.

In Figure 9a, the trajectory with p = 36 closely resembles the real trajectory, with a similar trajectory length and spacing between trajectory points. In Figure 9b, the trajectory with p = 36 is almost identical to the real trajectory, with the two trajectories overlapping closely. In Figure 9c, the trajectory with p = 30 closely follows the real trajectory, with a similar trajectory length. In Figure 9d, only the trajectory with p = 36 aligns with the direction of the real trajectory, while the directions of the other three trajectories deviate. However, the trajectory with p = 36 has a slightly longer trajectory length. It can be observed that trajectories with p = 48 are consistently longer in all scenarios and fail to capture the turning trend effectively. Trajectories with p = 42 and p = 30 show relatively similar trends to the real trajectory, but still exhibit some deviation. Overall, trajectories predicted with p = 36 are the most similar to the real trajectory.

4. Discussion

This paper proposes a time series prediction model, Informer-TP, which is based on historical AIS data of ships. Built upon the Informer model, this model enhances the ability to effectively predict future ship trajectories. Comparative analysis with other models demonstrates the effectiveness of Informer-TP in predicting ship trajectories.

Through a series of comparative experiments, we observed that the Informer-TP model demonstrates lower errors and higher accuracy in vessel trajectory prediction tasks. Specifically, when predicting 12, 18, and 24 points, the Informer-TP model reduced the HAV error by 26.74%, 16.34%, and 6.5%, respectively, compared to the LSTM-Seq2Seq model. Relative to the LSTM-Seq2Seq-Attention model, the HAV error reductions were 38.07%, 29.48%, and 10.37%, respectively. In comparison to the ConvLSTM-Seq2Seq model, the HAV error reductions were 47.53%, 39.56%, and 10.9%, respectively. Furthermore, the Informer-TP model outperformed the Transformer model, with HAV error reductions of 0.6%, 5.39%, and 5.04% for the respective prediction points. These results indicate that the

Informer-TP model is superior to the LSTM-Seq2Seq, LSTM-Seq2Seq-Attention, ConvLSTM-Seq2Seq, and Transformer models in terms of predictive accuracy. Moreover, we found that the length of L_{token} influences the prediction performance of the Informer-TP model. For a prediction length of 12, an L_{token} length of 24 yields the optimal performance. For a prediction length of 18, an L_{token} length of 42 is optimal, and for a prediction length of 24, an L_{token} length of 36 is most effective. However, other Seq2Seq models do not necessarily exhibit improved predictive performance with increased input length. Therefore, by increasing the input step length, the Informer-TP model can reduce prediction errors when forecasting trajectories of varying lengths. This adjustment also highlights the model's capability in long-sequence predictions.

The superior performance of the Informer-TP model can be attributed to its unique attention mechanism, which effectively extracts temporal features from trajectory data, thereby reducing prediction errors. Compared to several other models, it significantly enhances prediction accuracy. The prediction capability of the Transformer model is second only to the Informer-TP, possibly due to its powerful self-attention mechanism, but it is not as effective as the Informer-TP model in long-sequence prediction. The LSTM-Seq2Seq model excels in short-term predictions but shows a substantial increase in error for long-sequence predictions. On the other hand, the LSTM-Seq2Seq-Attention and ConvLSTM-Seq2Seq models demonstrate smaller error increases, indicating greater stability in long-sequence predictions, although they exhibit larger errors in short-term predictions. When the input step length is set to 42, these three models experience a sharp rise in errors, indicating their sensitivity to this parameter, which can adversely affect their prediction capabilities. The Informer-TP model, with its attention distillation mechanism, achieves higher prediction efficiency. To validate the prediction efficiency of the Informer-TP model, we used the test set to predict the vessel trajectory for the next 24 points and recorded the time required by each model. The results, as shown in Table 12, demonstrate that the Informer-TP model exhibits the lowest computation time, while the LSTM-Seq2Seq-Attention model takes the longest time.

Table 12. Computation time of different models when predicting 24 points.

Method	Prediction Time (s)
LSTM-Seq2Seq	9.0110
LSTM-Seq2Seq-Attention	35.2782
ConvLSTM-Seq2Seq	7.6746
Transformer	4.8682
Informer-TP	4.7496

In summary, by comparing with other baseline models and analyzing the impact of different L_{token} lengths on the model, it has been demonstrated that the Informer-TP model can effectively predict long-term ship trajectories. Furthermore, by comparing the prediction times of various models on the test set, it is shown that the Informer-TP model has a higher prediction efficiency. These experiments indicate that the Informer-TP model can efficiently and accurately predict long-term ship trajectories, consistent with the proposed conclusions. Compared to previous studies, this research provides a new method for long-term ship trajectory prediction. It significantly contributes to the development of maritime safety and intelligence.

5. Conclusions

For the long-term trajectory prediction of ships, this study proposes the method of Informer-TP for ship trajectory sequence prediction. To mitigate the impact of AIS data on prediction accuracy, this paper preprocesses AIS data using methods such as data cleaning, DBSCAN clustering, and cubic spline interpolation and divides trajectories into segments. Informer-TP is then used to predict ship trajectories, and compared with LSTM-Seq2Seq, LSTM-Seq2Seq-Attention, ConvLSTM-Seq2Seq, and Transformer models. The performance of the models with various output lengths is also explored.

Experimental results demonstrate that Informer-TP performs well in various scenarios, and with increasing prediction length, it exhibits the lowest prediction errors. Therefore, Informer-TP has the best overall performance. Compared with existing methods, the proposed Informer-TP method for ship trajectory prediction provides a new approach for ship trajectory planning, autonomous navigation, and risk avoidance, contributing to the safety and efficiency of ship navigation in complex marine environments.

5.1. Limitations

Although the prediction accuracy of the Informer-TP model used in this study is higher than that of the baseline model, the effectiveness of trajectory prediction for trajectories with frequently changing or significantly varying COG is still limited. Additionally, the model does not account for external factors such as weather conditions and interactions between ships.

5.2. Future Research

Future work should incorporate environmental information into the features to improve the accuracy of trajectory prediction. Additionally, we should explore models that account for interactions between ships to enhance the prediction accuracy for trajectories with frequently changing or significantly varying COG. These efforts aim to develop a more practical method for ship trajectory prediction.

Author Contributions: Conceptualization, C.X. and H.S.; methodology, C.X. and H.S.; software, H.S.; validation, C.X. and J.L.; investigation, C.X. and H.S.; data curation, C.X. and H.S.; writing—original draft, H.S.; writing—review and editing, C.X., X.W. and R.G.; visualization, H.S.; supervision, C.X and H.S.; project administration, C.X. and X.W.; funding acquisition, C.X. and X.W. All authors have read and agreed to the published version of the manuscript.

Funding: The research was supported by the National Natural Science Foundation of China (Grant Nos. 62002105 and 62201203), the Science and Technology Program of Hubei Province (Grant No. 2021BLB171), and the Chunhui Plan Collaborative Research Project, Ministry of Education, China (Grant No. HZKY20220350).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

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

References

1. Praetorius, G.; Lützhöft, M. Decision support for vessel traffic service (VTS): User needs for dynamic risk management in the VTS. *Work* **2012**, *41*, 4866–4872. [CrossRef]
2. Liu, H.; Jurdana, I.; Lopac, N.; Wakabayashi, N. BlueNavi: A microservices architecture-styled platform providing maritime information. *Sustainability* **2022**, *14*, 2173. [CrossRef]
3. Huai, S.; Zhang, S.; Wang, X.; Zhang, J. A Novel Adaptive Noise Resistance Method Used for AIS Real-Time Signal Detection. *Chin. J. Electron.* **2020**, *29*, 327–336. [CrossRef]
4. Vodas, M.; Pelekis, N.; Theodoridis, Y.; Ray, C.; Karkaletsis, V.; Petridis, S.; Miliou, A. Efficient ais data processing for environmentally safe shipping. *SPOUDAI J. Econ. Bus.* **2013**, *63*, 181–190.
5. Zhao, L.; Shi, G.; Yang, J. Ship trajectories pre-processing based on AIS data. *J. Navig.* **2018**, *71*, 1210–1230. [CrossRef]
6. Mou, J.M.; Van der Tak, C.; Ligteringen, H. Study on collision avoidance in busy waterways by using AIS data. *Ocean. Eng.* **2010**, *37*, 483–490. [CrossRef]
7. Rong, H.; Teixeira, A.; Soares, C.G. Data mining approach to shipping route characterization and anomaly detection based on AIS data. *Ocean. Eng.* **2020**, *198*, 106936. [CrossRef]
8. Xue, Y.; Liu, Y.; Xue, G.; Chen, G. Identification and prediction of ship maneuvering motion based on a Gaussian process with uncertainty propagation. *J. Mar. Sci. Eng.* **2021**, *9*, 804. [CrossRef]

9. Zhang, C.; Bin, J.; Wang, W.; Peng, X.; Wang, R.; Halldearn, R.; Liu, Z. AIS data driven general vessel destination prediction: A random forest based approach. *Transp. Res. Part C Emerg. Technol.* **2020**, *118*, 102729. [CrossRef]
10. Liu, J.; Shi, G.; Zhu, K. Vessel trajectory prediction model based on AIS sensor data and adaptive chaos differential evolution support vector regression (ACDE-SVR). *Appl. Sci.* **2019**, *9*, 2983. [CrossRef]
11. Zhou, H.; Chen, Y.; Zhang, S. Ship trajectory prediction based on BP neural network. *J. Artif. Intell.* **2019**, *1*, 29. [CrossRef]
12. Zhang, J.; Wang, H.; Cui, F.; Liu, Y.; Liu, Z.; Dong, J. Research into ship trajectory prediction based on an improved LSTM network. *J. Mar. Sci. Eng.* **2023**, *11*, 1268. [CrossRef]
13. Park, J.; Jeong, J.; Park, Y. Ship trajectory prediction based on bi-LSTM using spectral-clustered AIS data. *J. Mar. Sci. Eng.* **2021**, *9*, 1037. [CrossRef]
14. Fu, H.; Gu, Z.; Wang, H.; Wang, Y. Ship motion prediction based on ConvLSTM and XGBoost variable weight combination model. In Proceedings of the OCEANS 2022-Chennai, Chennai, India, 21–24 February 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1–8.
15. Han, P.; Wang, W.; Shi, Q.; Yang, J. Real-time short-term trajectory prediction based on GRU neural network. In Proceedings of the 2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC), San Diego, CA, USA, 8–12 September 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–8.
16. Jia, H.; Yang, Y.; An, J.; Fu, R. A ship trajectory prediction model based on attention-BILSTM optimized by the whale optimization algorithm. *Appl. Sci.* **2023**, *13*, 4907. [CrossRef]
17. Li, W.; Ren, J. Ship Roll Motion Prediction Using ConvLSTM with Attention Mechanism. In Proceedings of the 2022 41st Chinese Control Conference (CCC), Hefei, China, 25–27 July 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 5616–5620.
18. Bao, K.; Bi, J.; Gao, M.; Sun, Y.; Zhang, X.; Zhang, W. An improved ship trajectory prediction based on AIS data using MHA-BiGRU. *J. Mar. Sci. Eng.* **2022**, *10*, 804. [CrossRef]
19. Wu, W.; Chen, P.; Chen, L.; Mou, J. Ship Trajectory Prediction: An Integrated Approach Using ConvLSTM-Based Sequence-to-Sequence Model. *J. Mar. Sci. Eng.* **2023**, *11*, 1484. [CrossRef]
20. Billah, M.M.; Zhang, J.; Zhang, T. A Method for Vessel's Trajectory Prediction Based on Encoder Decoder Architecture. *J. Mar. Sci. Eng.* **2022**, *10*, 1529. [CrossRef]
21. Jiang, F.; Wang, H.; Li, Y. VesNet: A Vessel Network for Jointly Learning Route Pattern and Future Trajectory. *ACM Trans. Intell. Syst. Technol.* **2024**, *34*, 1–25. [CrossRef]
22. Lin, Z.; Yue, W.; Huang, J.; Wan, J. Ship trajectory prediction based on the TTCN-attention-GRU model. *Electronics* **2023**, *12*, 2556. [CrossRef]
23. Chen, J.; Chen, H.; Zhao, Y.; Li, X. FB-BiGRU: A Deep Learning model for AIS-based vessel trajectory curve fitting and analysis. *Ocean. Eng.* **2022**, *266*, 112898. [CrossRef]
24. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 6000–6010.
25. Kenton, J.D.M.W.C.; Toutanova, L.K. Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the NAACL-HLT 2019, Minneapolis, MN, USA, 2–7 June 2019; Volume 1, p. 2.
26. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2018, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141.
27. Zeng, A.; Chen, M.; Zhang, L.; Xu, Q. Are transformers effective for time series forecasting? In Proceedings of the AAAI Conference on Artificial Intelligence 2023, Washington, DC, USA, 7–14 February 2023; Volume 37, pp. 11121–11128. [CrossRef]
28. Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; Zhang, W. Informer: Beyond efficient transformer for long sequence time-series forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence 2021, Virtual, 2–9 February 2021; Volume 35, pp. 11106–11115.
29. Wu, H.; Xu, J.; Wang, J.; Long, M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 22419–22430.
30. Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; Jin, R. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In Proceedings of the International Conference on Machine Learning PMLR, 2022, Baltimore, MD, USA, 17–23 July 2022; pp. 27268–27286.
31. Liu, S.; Yu, H.; Liao, C.; Li, J.; Lin, W.; Liu, A.X.; Dustdar, S. Pyraformer: Low-Complexity Pyramidal Attention for Long-Range Time Series Modeling and Forecasting. In Proceedings of the International Conference on Learning Representations 2022, Virtual, 25 April 2022.
32. Nie, Y.; Nguyen, N.H.; Sinthong, P.; Kalagnanam, J. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. *arXiv* **2023**, arXiv:2211.14730.
33. Dong, L.; Wang, H.; Lou, J. An attention mechanism model based on positional encoding for the prediction of ship maneuvering motion in real sea state. *J. Mar. Sci. Technol.* **2024**, *29*, 136–152. [CrossRef]
34. Jiang, D.; Shi, G.; Li, N.; Ma, L.; Li, W.; Shi, J. TRFM-ls: Transformer-based deep learning method for vessel trajectory prediction. *J. Mar. Sci. Eng.* **2023**, *11*, 880. [CrossRef]
35. Zhao, W.; Wang, D.; Gao, K.; Wu, J.; Cheng, X. Large-Scale Long-Term Prediction of Ship AIS Tracks via Linear Networks with a Look-Back Window Decomposition Scheme of Time Features. *J. Mar. Sci. Eng.* **2023**, *11*, 2132. [CrossRef]

36. Nguyen, D.; Fablet, R. A Transformer Network with Sparse Augmented Data Representation and Cross Entropy Loss for AIS-based Vessel Trajectory Prediction. *IEEE Access* **2024**, *12*, 21596–21609. [CrossRef]
37. Qiang, H.; Guo, Z.; Xie, S.; Peng, X. MSTFormer: Motion Inspired Spatial-temporal Transformer with Dynamic-aware Attention for long-term Vessel Trajectory Prediction. *arXiv* **2023**, arXiv:2303.11540.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.