A Dynamic Context-Aware Approach for Vessel Trajectory Prediction Based on Multi-Stage Deep Learning

Xiaocai Zhang, Xiuju Fu, Zhe Xiao, Haiyan Xu, Wanbing Zhang, Jimmy Koh, and Zheng Qin

***Abstract*—A dynamic context-aware (DCA) approach based on multi-stage deep learning (MSDL), termed DCA-MSDL, is proposed to address this challenge. An inverted Transformer (iTransformer)-based deep learning architecture is first constructed for vessel turning status prediction. A deep generative framework is then presented for vessel trajectory augmentation. Following that, multiple potential trajectories are predicted using a novel data-driven algorithm based on multiple steps of nearest neighbor selection. Finally, trajectory enhancement considering dynamic traffic context is proposed to further improve prediction accuracy. With the separate steps of vessel turning status prediction, trajectory augmentation, trajectory prediction and trajectory enhancement, our approach allows us to explicitly explain the factors affecting the prediction accuracy and enable targeted improvements correspondingly. Extensive tests on real-world trajectories of vessels in the Singapore Strait have been conducted and the following encouraging results have been obtained: 1) the iTransformer-based turning status prediction achieves high accuracy at 93.37%, beating other state-of-the-art machine learning models; 2) the deep generative model-based trajectory augmentation reduces error by 8.26%, and it significantly outperforms other oversampling techniques; 3) DCA-MSDL increases the prediction accuracy by at least 37.24% in comparison to existing benchmarking methods; 4) the devised dynamic context-aware trajectory enhancement in DCA-MSDL significantly improves the accuracy by 3.53%.**

***Index Terms*—Maritime transport, context, vessel, turning status prediction, trajectory prediction, deep learning.**

I. Introduction

MARITIME transport supports approximately 90% of the global trade volume, being the most cost-efficient transport mode over long distances. With the rapid growth of the global economy in recent years, international trade volume has expanded significantly, resulting in an increase in the capacity, size and sailing speed of vessels [1]. Therefore, the safety and security of vessels have been elevated to an increasingly important level [2], especially in preventing collisions, which can lead to human fatalities, property and goods losses, and long-term environmental contamination. A tragic example of this was the collision between the SANCHI crude oil tanker and a bulk carrier vessel in the East China Sea, which caused the deaths of 32 crew members and the release or combustion of over 100,000 tons of petroleum products into the ocean [3].

Maritime collision avoidance systems (MCAS) play a critical role in preventing collisions by providing early alerts to nearby ships. A key aspect of these systems is predicting the trajectory of vessels, a task whose accuracy is vital for assessing the risk and determining further actions. The maritime and shipping sectors have increasingly adopted numerous sensors and systems, ushering in an age of big data with continuous collection of extensive maritime traffic information. This has significantly enhanced the systems' capabilities. Recently, the focus on vessel trajectory prediction (VTP) has intensified, making it a prominent area of research.

Autonomous ship technology is in the research and development (R&D) phase, with its implementation being a key focus. It promises to improve navigational safety for vessels. However, even with the advancements anticipated in the era of autonomous vessels, collision avoidance remains a significant hurdle in realizing these intelligent systems [4]. A crucial component in collision avoidance is the precise prediction of the trajectories of nearby vessels.

The widespread availability of AIS data has prompted a significant increase in research on VTP in recent years. According to the Safety of Life at Sea (SOLAS) convention in 2002, all ships above 300 gross tonnage which engage on international voyages are stipulated to equip with an automatic identification systems (AIS) transponder. Class A AIS transponder can transmit and receive all message types, including dynamic messages, semi-static and static messages. Dynamic messages are transmitted every 2 to 12 seconds, while semi-static and static messages are sent every 6 minutes [5]. Dynamic messages, such as position, speed and course, are updated on a regular basis automatically. Semi-static messages are manually entered and include voyage-related information, such as destination and estimated time of arrival (ETA). Static messages that are almost never altered include ship type, maritime mobile service identity (MMSI), etc. The static and dynamic messages of AIS data offer considerable support for the extensive research development of VTP.

Accurate VTP remains a challenge due to the complex, nonlinear, and stochastic nature of maritime transportation systems. To address this, we propose a multi-stage prediction approach to enhance the interpretability of the prediction, an

Manuscript received xxxxx. The Associate Editor for this article was xxxxx. (*Corresponding authors: Xiuju Fu and Zhe Xiao.*)

X. Zhang, X. Fu, Z. Xiao, X. Xu, W. Zhang and Z. Qin are with the Institute of High Performance Computing (IHPC), Agency for Science, Technology and Research (A*STAR), 1 Fusionopolis Way, #16-16 Connexis, Singapore 138632, Republic of Singapore. (e-mails: zhang_xiaocai@ihpc.a-star.edu.sg; fuxj@ihpc.a-star.edu.sg; xiaoz@ihpc.a-star.edu.sg; xuh@ihpc.a-star.edu.sg; zhangwab@ihpc.a-star.edu.sg; qinz@ihpc.a-star.edu.sg).

J. Koh is with the Pilotage and Digital Transformation Department, PSA Marine (Pte) Ltd, 70 West Coast Ferry Road, Singapore 126800, Republic of Singapore. (e-mails: jimmykoh@globalpsa.com).

aspect that has not received significant attention in previous studies. The way a ship moves when following a turning track compared to a straight track is markedly distinct, especially in waterways using the traffic separation scheme for separating traffic. Moreover, a ship making a turn may lead to unforeseen close-encounter situations in crowded water areas, requiring heightened vigilance. Additionally, the surrounding dynamic traffic context also influences the movement intentions of vessels, which has not been extensively investigated in previous studies. To tackle these challenges, first, we use an inverted Transformer (iTransformer)-based deep learning framework to predict a vessel's turning status. We then employ a generative adversarial network (GAN), a deep generative framework, to overcome sample size limitations for data-driven trajectory prediction, which enhances explainability and incorporates uncertainty into predictions. Finally, we improve the predicted trajectories by considering dynamic interactions between vessels and the surrounding traffic environment. Our prediction framework, termed DCA-MSDL (dynamic context-aware approach based on multi-stage deep learning), offers several **contributions**, which are summarized as follows:

1) This is the first VTP research from multi-stage perspectives to make the artificial intelligence (AI)-powered prediction more explainable, including turning status prediction, trajectory augmentation, trajectory prediction, and dynamic context-aware trajectory enhancement;
2) A large human-annotated data set for vessel turning status prediction is created in this study, and we first develop an iTransformer-based deep learning framework for addressing this problem;
3) We apply a GAN framework to synthesize large amounts of vessel trajectories for the follow-up data-driven trajectory prediction algorithm, which is able to deliver more accurate prediction while considering forecasting uncertainty;
4) The dynamic traffic context in the vessel's area of interest has been sensed and incorporated in our approach to further enhance the robustness and accuracy of trajectory prediction;
5) Extensive experiments have been conducted on the real-world trajectories of vessels in the Singapore Strait to confirm the effectiveness, robustness and superiority of DCA-MSDL in VTP.

II. Related work

In the following sections, we will provide a comprehensive review of existing VTP methods, which can be classified into four categories: simulation methods, statistical methods, machine learning methods, and hybrid methods [6].

A. Simulation methods

The simulation methods involve creating a digital version of a physical model from the real world [7]. This approach is straightforward but not commonly used on its own in VTP. Instead, it is often integrated with other methods like statistical and machine learning techniques, as seen in various hybrid methods [8]–[12]. For instance, Last *et al.* [13] developed a motion model without acceleration, using data from vessel trajectories. This model uses dynamic information such as latitude, longitude, course over ground (COG), speed over ground (SOG), and rate of turn (ROT). In the model, the geographical coordinates and COG are continuously updated, while SOG and ROT remain constant. However, this model may not always yield reliable predictions, especially when the intervals between trajectory data points are large, as uncertainty tends to increase over time.

B. Statistical methods

Statistical methods encompass methods based on neighborhood analysis, the Monte Carlo method (MCM), stochastic process (SP)-based techniques, methods utilizing Markov chains, and those based on filtering method.

The study by Hexeberg *et al.* [14] introduced a single point neighbor search (SPNS) method for VTP, focusing on estimating the course and speed of nearby vessels. Additionally, another study [15] presented a neighbor course distribution method (NCDM), capable of generating multiple trajectories while accounting for uncertainty. Alizadeh *et al.* [16] explored two types of similarity search methods using large-scale historical AIS data: one at the point level and the other at the trajectory level. The point-level similarity incorporates spatial, speed, and course factors, whereas the trajectory-level similarity employs dynamic time warping (DTW). Scheepens *et al.* [17] proposed a Monte Carlo-based model that differs from traditional approaches by simulating numerous synthetic trajectories to create a probability density field. Research by Uney *et al.* [18] and Millefiori *et al.* [19] introduced a hierarchical generative model using the Ornstein-Uhlenbeck (OU) process, designed for analyzing non-maneuvering motion characteristics and predicting vessel trajectories, showing efficacy in long-term forecasting with real-world data. Liu *et al.* [20] and Guo *et al.* [21] utilized a k-order multivariate Markov chain for long-term trajectory prediction, incorporating the location, speed, and course to construct the state-transition matrix. Zhang *et al.* [22] developed a wavelet-based hidden Markov chain (HMM) specifically for large ship VTP, transforming trajectory sequences into HMM input vectors via wavelet transform. The extended Kalman filter (EKF), a nonlinear version of the standard Kalman filter, was demonstrated to estimate position, speed, and acceleration of vessels [23], addressing the linear limitations of the Kalman filter. Furthermore, the particle filter (PF) method was also applied to this field, with Lian *et al.* [24] employing PF to predict AIS trajectories, particularly focusing on updating the gaps in AIS data caused by information delays.

C. Machine learning methods

The machine learning (ML) methods in VTP can be generally dichotomized into two families: traditional ML methods and recent deep learning (DL) methods.

For traditional ML, unsupervised ML algorithms like dimensionality reduction, clustering as well as other supervised learning methods have been involved in this research. Murray

et al. [25] utilized principal component analysis (PCA) to extract feature vectors for trajectory analysis, which were then fed into a Gaussian mixture model (GMM) clustering algorithm for potential route generation. Pallotta *et al.* [26] introduced an incremental density-based spatial clustering of applications with noise (DBSCAN) method for identifying waypoints over specific sea areas, aiding in recognizing vessel movement patterns and enhancing route prediction. Regarding supervised learning methods, various techniques have been employed, including k-nearest neighbors (k-NN) [27], support vector machine (SVM) [28], [29], artificial neural network (ANN) [30], and extreme learning machine (ELM) [31]. The k-NN is used for multi-class classification problems, representing coordinates as discrete grid cells [27]. The SVM in Liu *et al.* [29] was enhanced with an adaptive chaos differential evolution method for optimizing parameters. Additionally, Liu *et al.* [28] developed a least-squares SVM, tuning parameters using particle swarm optimization. In [30], the ANN method was implemented with the Levenberg-Marquardt (LM) algorithm for parameter updates. Tu *et al.* [32] proposed an ensemble learning approach based on ELM, where each data cluster is processed by a single ELM to train a model, followed by a weighted k-NN method for model selection during the forecasting phase. Li *et al.* [33] conducted a thorough comparison of five different machine learning models, including KF, SVR, back-propagation network (BP), Gaussian process regression (GPR), and random forest, to assess their performance in predicting ship trajectories. The findings revealed that the order of effectiveness for these machine learning methods was SVR > KF > GPR > RF > BP.

The DL methods include recurrent neural network (RNN) (i.e., long short-term memory (LSTM) and gated recurrent unit (GRU)), encoder-decoder architecture, deep neural network (DNN), convolutional neural network (CNN) and transformer neural network. RNN and its variants are the most frequently used approaches for predicting vessel trajectory.

Tang *et al.* [34] developed an LSTM model for predicting vessel trajectories in Tianjin Port, China, using a two-layer LSTM structure and inputting 10-minute vessel state observations. Recognizing the impact of vessel type on movement, Mehri *et al.* [35] trained distinct LSTM models for different vessel types. A Bi-LSTM network was introduced in [36], offering improved historical and future data correlation over single-directional LSTM. Wang *et al.* [37] enhanced Bi-LSTM with an attention mechanism for selective focus on hidden information segments [38]. A novel variational LSTM approach in maritime trajectory prediction was first proposed in [39], demonstrating its ability to learn additional latent features compared to standard LSTM. Several studies [40]–[44] also utilized LSTM-based encoder-decoder architectures. GRU, a simpler alternative to LSTM, was used in two studies [45] for vessel trajectory forecasting. Xiao *et al.* [46] created a novel LSTM and Bi-LSTM-based encoder-decoder network, enhancing learning from vessel trajectories in both directions. Xu *et al.* [47] built a deep architecture using stacked Bi-GRU layers, while another study [45] presented a GRU-based encoder-decoder architecture, showing superior short-term prediction capabilities compared to LSTM and GRU models. An attentional Bi-LSTM and LSTM-based recurrent encoder-decoder network for VTP was highlighted in [48]. A DNN framework was discussed for general-purpose merchant vessel trajectory prediction [49], although it faced overfitting issues, prompting suggestions for bio-inspired model integration for accuracy improvements. Kim *et al.* [50] proposed a CNN framework for extracting features from comprehensive ship movement data. A transformer neural network for trajectory prediction was presented by Nguyen *et al.* [51], embedding sequential inputs into higher-dimensional discrete vectors and functioning as a classifier to predict attribute distributions. Li *et al.* [33] conducted a systematic comparison of seven deep learning models, including RNN, LSTM, GRU, Bi-LSTM, Seq2seq, Bi-GRU, and transformer, for short-term ship trajectory prediction, ranking their effectiveness as Bi-GRU > GRU > transformer > Bi-LSTM > LSTM > Seq2seq > RNN.

D. Hybrid methods

The hybrid methods incorporate multiple methods to enhance performance by leveraging the advantages of different methods.

In their research, the authors of [52] suggested a model combining clustering and a multiple trajectory extraction method (MTEM), an advancement over the SPNS method previously mentioned. MTEM differentiates from Hexeberg *et al.* [14] by clustering points within a similar Vincenty distance, followed by further clustering of AIS data based on COG and SOG. The model then eliminates duplicate points from the same vessel to maintain a single point per vessel in each cluster. It concludes with a prediction algorithm using the average COG and SOG from the clustered data. Murray *et al.* [53] constructed a hybrid framework consisting of trajectory clustering, classification, and prediction. This framework starts with GMM for clustering, uses k-NN for classifying trajectories into clusters, and then applies a dual linear autoencoder for trajectory prediction within the selected cluster. A multi-stage prediction model was introduced by [8], involving support point prediction, destination point prediction, and trajectory interpolation. This model uses an LSTM model for support point prediction and historical data filtering for destination prediction, assuming certain predefined conditions. The trajectory is then simulated using cubic spline interpolation based on the support point and destination data. Suo *et al.* [54] developed a hybrid model that combines unsupervised clustering (DBSCAN) and deep learning (GRU). Before training with the GRU model, DBSCAN is applied to trajectory data to identify the main vessel trajectory zones, aiming to remove redundant data through similarity analysis. In [9], a hybrid framework for VTP in Singapore was proposed. It begins with a multi-layer perceptron for predicting COG and SOG, followed by motion modeling to determine vessel positions based on these parameters. The PF method is then employed to correct the COG sequence when a movement pattern is detected in the historical knowledge base.

III. METHODOLOGY

The workflow of our DCA-MSDL is depicted in Fig. 1. It contains four modules: vessel turning status prediction, trajectory augmentation, data-driven trajectory prediction, and dynamic context-aware trajectory enhancement. Given the latest available trajectory data (for example t_1 m) of the vessel to be predicted, the navigation status of the vessel is predicted first with an interval of t_2 m (t_2 could be 0.5, 1, 2, ...). Once a "turning" output is captured, the trajectory prediction module will be executed every t_3 m (e.g., $t_3 = 1$). If the vessel is predicted as "going straight", the turning status will be predicted continuously until a "turning" status is captured. Details about methodologies be provided in this section, along with some initial definitions and concepts.

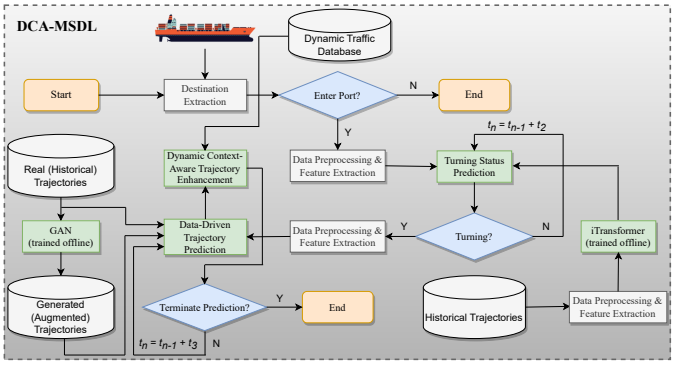


Fig. 1. The workflow of DCA-MSDL. iTransformer and GAN are both trained offline. The turning status prediction, trajectory prediction, and trajectory enhancement are performed in real time. The vessel's turning status is detected every 30 s, while the trajectory prediction and enhancement are carried out every 1 m. The trajectory prediction algorithm will be terminated when it has met the defined terminal condition, e.g., the vessel has passed through the traffic separation zone.

A. Definitions

Definition 1: Trajectory. A trajectory is defined as a sequence of tuples $\{x_t, t \in T\}$, $x_t = (l_t, o_t, \cdots)$, where x_t is consisted of geo-locations l_t or other attributes o_t (e.g., speed, course, etc.). l_t is a sequence of points in 2-dimensional space, $l_t = \{(\varphi_1, \lambda_1), (\varphi_2, \lambda_2), \cdots, (\varphi_t, \lambda_t)\}$. T is a set of timestamps with equidistant time intervals, and $T = \{1, 2, 3, \cdots\}$. φ and λ stands for latitude and longitude, respectively.

Definition 2: Turning. Turning refers to the process of a vessel significantly changing her course over ground (COG) towards the channel.

Definition 3: Turning Status. A movement status of a vessel at timestamp t, denoted as s_t, indicates whether the vessel is going straight or turning, where $s_t \in \{lb_{tr}, lb_{gs}\}$, lb_{tr} means "turning" status and lb_{gs} denotes "going straight" status.

Definition 4: Turning Status Prediction. Given a fully observed trajectory at timestamps $\{1, 2, \cdots, t\}$ denoted as $\mathbf{X} = \{x_1, x_2, \cdots, x_t\}$, the aim is to predict a vessel's turning status at timestamp t, denoted by $\mathbf{Y} = s_t$.

Definition 5: Trajectory Prediction. Given a fully observed trajectory at timestamps $\{1, 2, \cdots, t\}$ denoted as $\mathbf{X} = \{x_1, x_2, \cdots, x_t\}$, the objective is to predict a vessel's trajectories at timestamp $t+1, t+2, \cdots, t+b$, denoted by $\mathbf{Y} = \{x_{t+1}, x_{t+2}, \cdots, x_{t+b}\}$, where b stands for the prediction length.

B. iTransformer-based turning status prediction

The iTransformer is an adaptation of the standard Transformer model, specifically designed to improve its performance in time-series analysis [55]. Unlike the traditional Transformer that combines multiple data points from the same time instance into single channels, the iTransformer processes each time series of a variable separately as individual tokens. It primarily utilizes the Transformer's encoder component and employs self-attention for these distinct variate tokens, thereby improving interpretability and uncovering correlations among multiple variables. Fig. 2 illustrates the comparison between the vanilla Transformer and the iTransformer in time-series representation learning [55].

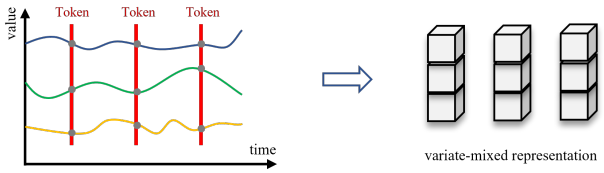
The encoder E transforms the input time-series into a sequence of time-series embeddings. Assume we have N layers of transformer in the encoder, this can be represented as

$$E\left(S;\theta_E\right) = E_N\left(\cdots E_1\left(S;\theta_{E_1}\right);\theta_{E_N}\right), \tag{1}$$

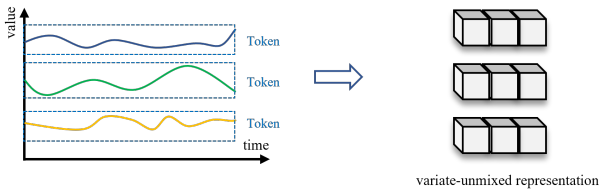
where θ_{E_i} denotes the parameters of the ith layer of the transformer. The encoder uses transformer architecture which is mainly based on a self-attention mechanism. The self-attention mechanism in each layer can be described as

$$A\left(Q,K,V\right) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{2}$$

where Q, K, and V are the query, key, and value matrices respectively, and d_k is the dimension of the key vectors.



(a) Transformer



(b) iTransformer

Fig. 2. Comparison between the Transformer (a) and iTransformer (b): While the Transformer integrates a temporal token that encapsulates the multivariate representation at each time step, the iTransformer takes a different approach by embedding each series separately into a variate token.

Suppose at timestamp $t+m$, we have the historical trajectory at timestamps $(t+1, t+2, \cdots, t+m)$, the task is to predict the vessel's turning state at timestamp $t+m$. The input $\mathbf{X}$ of the model is represented by

$$\mathbf{x}_j = ((\varphi_{j1}, \lambda_{j1}, v_{j1}, \theta_{j1}, d_{j1}), (\varphi_{j2}, \lambda_{j2}, v_{j2}, \theta_{j2}, d_{j2}), \cdots, (\varphi_{jm}, \lambda_{jm}, v_{jm}, \theta_{jm}, d_{jm}))^T \in \mathbb{R}^{5\times m}, \tag{3}$$

and

$$\mathbf{X} = \left(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{N-1}, \mathbf{x}_N\right) \in \mathbb{R}^{N \times m \times 5}, \tag{4}$$

where φ, λ, v, θ, and d stands for the latitude, longitude, speed, course and distance to the traffic separation zone, m is the sequence length, N is the number of observations.

$$y_{rep} = f_{ST5}\left(X; \theta_E\right), \tag{5}$$

where f_{ST5} signifies the Sentence-T5 model that has been pre-trained. Following that, y_{pre} is connected to a fully-connected (FC) layer, as formulated by

$$y_{FC} = f_{FC}\left(y_{pre}; \theta_{FC}\right), \tag{6}$$

where θ_{MLP} stands the weights of FC layer. Following that, a Softmax function ($\sigma_{Softmax}$) is attached to get probability distribution with a nonlinear transformation, as indicated by

$$\mathbf{y} = \sigma_{Softmax}(\mathbf{y}). \tag{7}$$

Let $\hat{\mathbf{y}}_j$ denote the ground-truth class label (i.e., going straight or turning), and $\mathbf{y}_j$ signify the corresponding outputted class probability distribution from iTransformer; thus, the loss (i.e., cross entropy-loss) can be derived according to

$$\mathcal{L}(\mathbf{W}, \mathbf{b}) = -\sum_{j=1}^{N}\left(\hat{\mathbf{y}}_j \log\left(\mathbf{y}_j\right) + \left(1 - \hat{\mathbf{y}}_j\right)\log\left(1 - \mathbf{y}_j\right)\right), \tag{8}$$

where N means the sample size, and $\mathbf{W}$ and $\mathbf{b}$ stand for the neural network's weights and biases, respectively.

The goal of network training is to optimize the weights and biases using the training data, aiming to minimize the total loss function. This is represented as

$$(\mathbf{W}^*, \mathbf{b}^*) = \arg\min_{\mathbf{W},\mathbf{b}} \mathcal{L}(\mathbf{W}, \mathbf{b}), \tag{9}$$

where $\mathcal{L}(\mathbf{W}, \mathbf{b})$ denotes the overall loss function described in Eq. (8). This process closely resembles standard deep neural network training. The optimal parameters $(\mathbf{W}^*, \mathbf{b}^*)$ are determined using the commonly employed back-propagation algorithm with an Adam optimizer.

C. GAN-based trajectory augmentation

To address the challenge of insufficient trajectory for the subsequent data-driven prediction, a GAN-based data augmentation technique is dedicated to augmenting trajectories that cover diverse scene-consistent motion modes [56]. The whole architecture of the GAN model for trajectory augmentation is shown in Fig. 3. A basic GAN framework consists of a generator (G) and a discriminator (D), which are trained simultaneously [57]. The G is to capture the data distribution, while D is to estimate the probability of a sample being come from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake, while for D is to minimize the discriminative error. In this study, the training data is represented as

$$\mathbf{T} = \left(\mathbf{t}_1, \mathbf{t}_2, \cdots, \mathbf{t}_N\right) \in \mathbb{R}^{N \times L \times 3}, \tag{10}$$

and

$$\mathbf{t}_i = ((\varphi_{i,1}, \lambda_{i,1}, v_{i,1}), \cdots, \varphi_{i,L}, \lambda_{i,L}, v_{i,L})) \in \mathbb{R}^{L \times 3}, \tag{11}$$

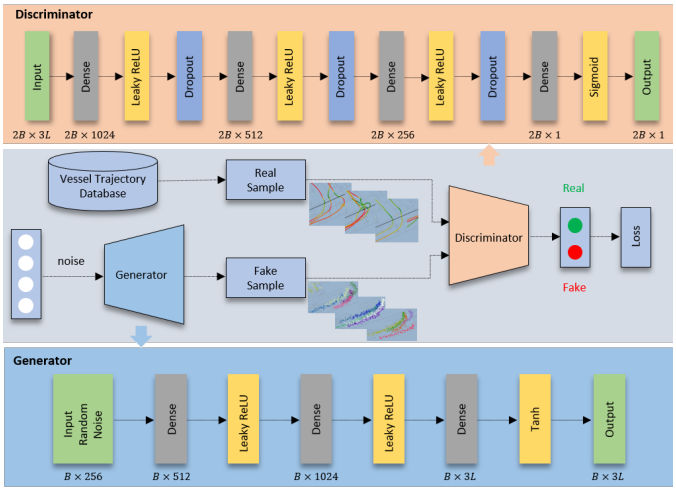


Fig. 3. The structure of GAN for vessel trajectory augmentation. It comprises a generator and a discriminator. The structures of the generator and discriminator are displayed in the upper and lower diagrams, respectively. B denotes the batch size. L is the sequence length.

where N is the sample size, L stands for the sequence length, including the input length ($\frac{L}{2}$) and prediction length ($\frac{L}{2}$). In this study, $L = 120$. The input length and prediction length are all set as 60. 3 is the dimension of the feature, where three features are selected: latitude φ, longitude λ, and speed v.

1) Generator: The generator is based on a multi-layer perceptron (MLP) neural network, the architecture of which is shown in the lower diagram of Fig. 3. The dimensions for the input, output, as well as each hidden layer, have also been provided for a better understanding of our model. Suppose the random noise denotes as $\mathbf{z}$, it is randomly generated between -1 and 1 with a uniform probability distribution, as shown by

$$\mathbf{z} = \text{uniform}\left(-1, 1, \text{size} = (B, 256)\right) \in \mathbb{R}^{B \times 256}, \tag{12}$$

where B stands for the batch size for network training via a mini-batch gradient descent algorithm. The input random noise is then transformed by employing three dense layers, two leaky ReLU layers, and one Tanh layer. The transformation processes of the generator are formulated as follows:

$$\mathbf{f}_{g1} = F_{dense}\left(\mathbf{z}; \mathbf{W}_{g1}\right), \tag{13}$$

$$\sigma_{LeakyReLu}(x) = \max\left(\alpha x, x\right), \tag{14}$$

$$\mathbf{f}_{g2} = \sigma_{LeakyReLu}\left(\mathbf{f}_{g1}\right), \tag{15}$$

$$\mathbf{f}_{g3} = F_{dense}\left(\mathbf{f}_{g2}; \mathbf{W}_{g3}\right), \tag{16}$$

$$\mathbf{f}_{g4} = \sigma_{LeakyReLu}\left(\mathbf{f}_{g3}\right), \tag{17}$$

$$\mathbf{f}_{g5} = F_{dense}\left(\mathbf{f}_{g4}; \mathbf{W}_{g5}\right), \tag{18}$$

and

$$\mathbf{f}_{g\_oup} = \sigma_{Tanh}\left(\mathbf{f}_{g5}\right). \tag{19}$$

The final output $\mathbf{f}_{g\_oup}$ of the generator is then derived, and this will be used as an input source for the discriminator that follows.

2) Discriminator: The implementation of the discriminator is depicted in the upper diagram of Fig. 3. In fact, the discriminator is performed as a binary classifier to distinguish whether the sample is a real trajectory or a fake trajectory generated by the G. As a result, the outputs from the generator are all labeled as the class of fake sample, as shown by

$$\mathbf{T}_{fake} = \mathbf{f}_{g\_oup}. \tag{20}$$

Accordingly, the real samples $\mathbf{T}_{real}$ are from the training data $\mathbf{T}$, which are scaled into $[-1, 1]$ range, and then reshaped into the same dimensional space as $\mathbf{f}_{g\_oup}$ (i.e., $B \times 3L$), as formulated by

$$\mathbf{T}_{real} \subseteq \mathbf{T}, \tag{21}$$

and

$$\mathbf{T'}_{real} = F_{reshape}\left(\mathbf{T}_{real}, \text{size} = (B, 3L)\right). \tag{22}$$

Following that, the real trajectories $\mathbf{T'}_{real}$ and fake trajectories $\mathbf{T}_{fake}$ are concatenated to form the input $\mathbf{x}$ of discriminator using

$$\mathbf{x} = F_{concat}\left(\mathbf{T}_{fake}, \mathbf{T'}_{real}\right) \in \mathbb{R}^{2B \times 3L}, \tag{23}$$

then, $\mathbf{x}$ is transformed via four dense layers, three leaky ReLU layers, and three dropout layers. The latent feature is represented as $\mathbf{f}_{d10}$. Finally, a sigmoid activation layer is appended to map the latent feature between 0 and 1. The procedure of discriminator is formulated as

$$\mathbf{f}_{d1} = F_{dense}\left(\mathbf{x}; \mathbf{W}_{d1}\right), \tag{24}$$

$$\mathbf{f}_{d2} = \sigma_{LeakyReLu}\left(\mathbf{f}_{d1}\right), \tag{25}$$

$$\mathbf{f}_{d3} = F_{dropout}\left(\mathbf{f}_{d2}, \delta\right), \tag{26}$$

$$\mathbf{f}_{d10} = F_{dense}\left(\mathbf{f}_{d9}; \mathbf{W}_{d10}\right), \tag{27}$$

and

$$\mathbf{f}_{d\_oup} = \sigma_{Sigmoid}\left(\mathbf{f}_{d10}\right), \tag{28}$$

where δ denotes the dropout rate.

3) Loss function: The loss function of GAN is formulated as

$$\mathcal{L} = -\mathbb{E}_x\left[\log\left(D\left(x\right)\right)\right] - \mathbb{E}_z\left[\log\left(1 - D\left(G\left(z\right)\right)\right)\right], \tag{29}$$

where x is the real instance, $D(x)$ is the discriminator's probability estimate of x. $\mathbb{E}_x$ is the expected value over all real samples. $G(z)$ is the generator's output when given noise z. $D(G(z))$ is the discriminator's estimate of the probability that a fake instance is real. $\mathbb{E}_z$ is the expected value over all random inputs to the generator.

The generator G tries to maximize the loss function $\mathcal{L}$, while the discriminator D attempts to minimize it. The overall objective function L^* is

$$\mathcal{L}^* = \max_G \min_D \mathcal{L}. \tag{30}$$

D. Trajectory prediction

With the well-trained generator G of the GAN framework, we are able to synthesize more possible trajectories. Suppose the generated trajectories is $\mathbf{T}_{generate}$, we have

$$\mathbf{T}_{generate} \in \mathbb{R}^{S \times 3L}, \tag{31}$$

and

$$\mathbf{T}_{generate}{}' = F_{reshape}\left(\mathbf{T}_{generate}, \text{size} = (S, L, 3)\right), \tag{32}$$

where S means the sample size. Next, we devise a trajectory prediction algorithm based on $\mathbf{T}_{generate}{}'$ and $\mathbf{T}$. The algorithm comprises four steps: trajectory fitting, neighbors pre-selection, trajectory shifting, and neighbors final selection.

1) Trajectory fitting: Since the generated trajectories $\mathbf{T}_{generate}{}'$ may contain some noisy coordinates, a data fitting procedure is required to obtain a continuous and smooth trajectory. Data fitting is only applied to the latitudinal and longitudinal dimensions of the generated trajectories. We attempt to utilize the following nonlinear equation to fit these two features,

$$f(x) = c_0 + c_1 x + c_2 x^2 + c_3 x^3 + c_4 x^4 + c_5 \sin\left(c_6 x + c_7\right), \tag{33}$$

where $c_0, c_1, \cdots, c_7$ are the parameters for estimation. The least squares method is adopted for parameter estimation.

2) Neighbors pre-selection: We define the trajectories via fitting as $\mathbf{T}^{s1}_{generate}$, which is derived by

$$\mathbf{T}^{s1}_{generate} = F_{fitting}\left(\mathbf{T}_{generate}{}'\right), \tag{34}$$

we then combine it with the training trajectories $\mathbf{T}$ to enlarge the data set for a pre-selection in this step, as shown by

$$\mathbf{T}^{s1}_{generate} = F_{concat}\left(\mathbf{T}^{s1}_{generate}, \mathbf{T}\right). \tag{35}$$

In this step of pre-selection, the nearest neighbors are identified based on only two indices: latitude and longitude. We roughly select k_1 nearest neighbors from $\mathbf{T}^{s1}_{generate}$, where k_1 is suggested to be a large number. The input sequence trajectories with the length of $\frac{L}{2}$ are extracted to calculate the positional similarity defined by the DTW formulation, as shown by

$$sim_{position}(q, g) = \text{DTW}\left(q, g\right), \tag{36}$$

where q is the query trajectory, and g is the generated trajectory from $\mathbf{T}^{s1}_{generate}$. Assume the pre-selected trajectories denote as $\mathbf{T}^{s2}_{generate}$, it can be written as

$$\mathbf{T}^{s2}_{generate} = F_{pre\_selection}\left(\mathbf{T}^{s1}_{generate}, k_1\right). \tag{37}$$

3) Trajectory shifting: The shifting of trajectory is to guarantee that the reference point in each pre-selected trajectory is consistent with the current position of the vessel. Fig. 4 provides a simple example of the shifting process of a trajectory. The orange trajectory is a pre-selected trajectory, symbolized as $\mathbf{T}_P$, and $\mathbf{T}_P \in \mathbf{T}^{s2}_{generate}$. The black trajectory stands for a query trajectory, represented as $\mathbf{T}_Q$. The red point Q denotes the current location of a vessel, and the blue position P means the reference position (i.e., the middle position of the trajectory in this study) of a generated trajectory. The latitude

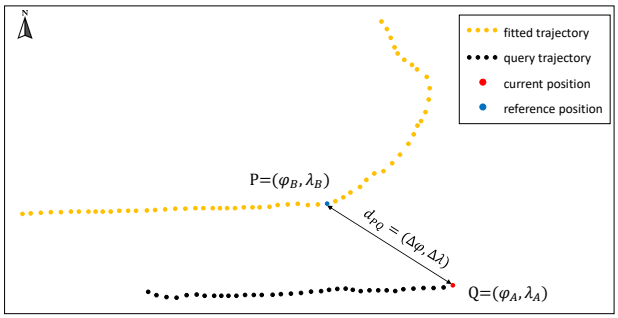


Fig. 4. Illustration of an example of trajectory shifting.

displacement $\Delta\varphi$ and longitude displacement $\Delta\lambda$ between Q and P are derived by

$$\Delta\varphi = \varphi_Q - \varphi_P, \tag{38}$$

and

$$\Delta\lambda = \lambda_Q - \lambda_P. \tag{39}$$

Following that, all the latitudes and longitudes of the trajectory are translated based on the above two displacements. For arbitrary latitude $\varphi_{i,P}$ and longitude $\lambda_{i,P}$ of trajectory $\mathbf{T}_P$, its updated values ($\varphi'_{i,P}$ or $\lambda'_{i,P}$) are obtained via

$$\varphi'_{i,B} = \varphi_{i,B} + \Delta\varphi, \tag{40}$$

and

$$\lambda'_{i,B} = \lambda_{i,B} + \Delta\lambda. \tag{41}$$

In a similar way, the procedure of trajectory shifting can be represented by the following equation,

$$\mathbf{T}^{s3}_{generate} = F_{shifting}\left(\mathbf{T}^{s2}_{generate}, \mathbf{T}_Q\right), \tag{42}$$

where $\mathbf{T}_Q$ is the query trajectory.

4) Neighbors final selection: In this phase of selection, we incorporate the information of vessel speed. As the location and speed exhibit varied impacts on the quality of selection, and we do not have much prior knowledge about it, we introduce a two-step selection mechanism to select the most potential candidates based on location and speed successively.

In the second round of selection, we only use the most recent l speed records ($l < \frac{L}{2}$) with the consideration that vessels are more likely to adjust their speeds before or during entering port, and the most recent of such information is more important and relevant to link the future trajectory. The selection of this step is conducted only on those k_1 trajectories from the last step. The equation for the second step selection is formulated by

$$sim_{speed}(q,g) = \sqrt{\sum_{i=1}^{l}(v_{i,q} - v_{i,g})^2}. \tag{43}$$

Similarly, the final k_2 nearest neighbors selected based on Eq. (43) are regarded as the most potential predicted trajectories for a given query trajectory. The final k_2 trajectories are derived by

$$\mathbf{T}^{s4}_{generate} = F_{final\_selection}\left(\mathbf{T}^{s3}_{generate}, k_2, l\right). \tag{44}$$

E. Dynamic context-aware (DCA) trajectory enhancement

As the navigation of vessels in waterways can be viewed as an interactive system between agent and environment, we assume that the movement behaviors of vessels may be affected by the surrounding traffic environment. Therefore, the derived trajectories $\mathbf{T}^{s4}_{generate}$ could be further enhanced by quantifying the impact of the traffic environment on the vessel. For example, the vessel may revocate her turning intention once the crew member foresees the potential collision risk with such a turning procedure. As a result, we have incorporated such a scenario into our prediction model to make the prediction more reasonable and explainable. The collision risk between two vessels is measured by the distance to the closest point of approach (DCPA). A lower DCPA indicates a higher risk of collision between the two vessels. Fig. 5 briefly depicts how we measure the DCPA of two vessels (i.e., vessels A and B). Vessel A is the target vessel to be predicted with trajectory, and she is currently conducting the turning maneuver. Vessel B is a nearby vessel that is currently presented in the area of interest of vessel A. ● denotes the current positions of vessels, and the predicted trajectory using our method is color-coded by ● (from t_1 to t_n). We assume that vessel B keeps her course and speed, and based on this assumption, her possible trajectory then can be inferred linearly, as represented by ● (from t_1 to t_n). For arbitrary timestamp t_i, $0 \leqslant i \leqslant n$, we have

$$d_{t_i} = \text{distance}\left(A_{t_i}, B_{t_i}\right), \tag{45}$$

and

$$\text{DCPA}_{AB} = \min\left(d_{t_1}, d_{t_2}, \cdots, d_{t_n}\right), \tag{46}$$

where the distance function in Eq. (45) employs the Haversine distance in this study.

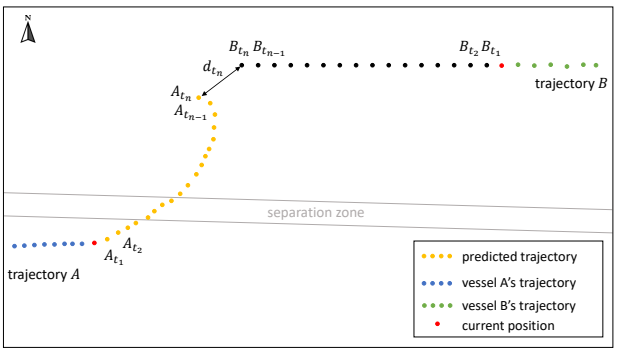


Fig. 5. Calculation of the DCPA between vessel A and vessel B.

All the pair-wise DCPAs of vessel A are derived by scanning all the vessels (such as vessels $B, C, \cdots$) in her area of interest, and the minimum DCPA (denoted as MDCPA) is obtained by

$$\text{MDCPA}_A = \min\left(\text{DCPA}_{AB}, \text{DCPA}_{AC}, \cdots\right). \tag{47}$$

If MDCPA_A is less than a preset threshold ϵ (unit in nautical mile (nm)), we suppose that the collision risk is high, and the turning maneuver should be avoided. Instead of turning, the vessel keeps going straight with constant speed and course.

Details of the DCA-MSDL algorithm for VTP are explained in Algorithm 1. The iTransformer and GAN models are both trained offline (steps 1 and 2). A chunk of generated trajectories with the size of S is produced via GAN with the random noise as input (steps 3 and 4). Following that, reshaping, fitting, and concatenation are performed on the generated trajectories, successively (steps 5 to 7). If the iTransformer model predicts that the vessel will turn, the trajectory prediction algorithm will be carried out in real time, as illustrated in steps 9 to step 19. Otherwise, the turning status prediction will continue for the next timestamp prediction (step 20). The dynamic context-aware trajectory enhancement is described in steps 13 to 18, where the high-risk and low-risk scenarios are formulated in steps 15 and 17, respectively.

Algorithm 1 DCA-MSDL algorithm for VTP

Parameters: S, L, k_1, k_2, l, ϵ
Input: $\mathbf{T}_Q$, $\mathbf{T}_{real}$
Output: $\mathbf{T}_{enhanced}$
1: $iTransformer \leftarrow$ *train an iTransformer model offline*;
2: $GAN \leftarrow$ *train a GAN model offline*;
3: *random* $\mathbf{z}$ *with size of* S;
4: $\mathbf{T}_{generate} \leftarrow GAN(S, \mathbf{z})$;
5: $\mathbf{T}_{generate}{}' \leftarrow F_{reshape}\left(\mathbf{T}_{generate}, size = (S, L, 3)\right)$;
6: $\mathbf{T}^{s1}_{generate} \leftarrow F_{fitting}\left(\mathbf{T}_{generate}{}'\right)$;
7: $\mathbf{T}^{s1}_{generate} \leftarrow F_{concat}\left(\mathbf{T}^{s1}_{generate}, \mathbf{T}\right)$;
8: $y_{iTransformer} \leftarrow iTransformer(\mathbf{T}_Q)$;
9: **if** $y_{iTransformer}$ *is "turning"* **then**
10: $\quad\mathbf{T}^{s2}_{generate} \leftarrow F_{pre\_selection}\left(\mathbf{T}^{s1}_{generate}, k_1\right)$;
11: $\quad\mathbf{T}^{s3}_{generate} \leftarrow F_{shifting}\left(\mathbf{T}^{s2}_{generate}, \mathbf{T}_Q\right)$;
12: $\quad\mathbf{T}^{s4}_{generate} \leftarrow F_{final\_selection}\left(\mathbf{T}^{s3}_{generate}, k_2, l\right)$;
13: $\quad$MDCPA $\leftarrow$ *scan nearby vessels in area of interest and calculate the minimum* DCPA
14: $\quad$**if** MDCPA $< \epsilon$ **then**
15: $\quad\quad\mathbf{T}_{enhanced} \leftarrow$ *going straight with constant speed and course*;
16: $\quad$**else**
17: $\quad\quad\mathbf{T}_{enhanced} \leftarrow \mathbf{T}^{s4}_{generate}$;
18: $\quad$**end if**
19: **else**
20: $\quad\mathbf{T}_Q \leftarrow$*update* $\mathbf{T}_Q$ *for prediction at the next timestamp*;
21: **end if**
22: **return** $\mathbf{T}_{enhanced}$

IV. Experiments and analyses

We have performed extensive experiments in this section to answer the following research questions:

RQ1: How accurately does our proposed iTransformer algorithm achieve in the turning status prediction task? And how well does it compare with other machine learning models?

RQ2: Is the formulated approach of GAN effective for trajectory augmentation? And how well does it in comparison with other data augmentation techniques?

RQ3: Why do we propose the method described in Section III-D for VTP rather than adopting machine learning methods? What are the advantages of our devised data-driven trajectory prediction algorithm?

RQ4: How significantly does the trajectory enhancement based on dynamic traffic context improve the prediction?

RQ5: What is the effect of different steps in the developed trajectory prediction algorithm? How well does our DCA-MSDL perform in contrast to other existing baseline methods in the VTP problem?

A. Data sets

A real-world AIS data set collected in the Singapore Strait is selected as the experimental data. The AIS data spans 14 months, from 1 January 2017 to 28 February 2018. Only the vessels entering the Port of Singapore are selected for experiments, as the vessels in this scenario will perform turning operations, which makes trajectory prediction more difficult because of increasing uncertainty. We use the last two months' AIS data, i.e., January and February of 2018, as the test data, and the rest as the training data. The interpolation time interval for trajectory data preprocessing is 10 s. After trajectory interpolation, a total of 7,828 training samples and 1,026 test samples are acquired for the turning status prediction task. All the labels (e.g., "turning" or "going straight") in this data set are manually annotated by ourselves. Meanwhile, 16,843 training samples and 3,700 test samples are obtained for the tasks of trajectory augmentation as well as trajectory prediction.

B. Evaluation metrics

For the turning status prediction task, we employ metrics such as accuracy (Acc.), precision (Pre.), recall (Rec.), F1-score, and area under the ROC curve (AUC) to evaluate the performance. For assessing the quality of prediction, we utilize a mean position error (MPE) based on Haversine distance, which is described as

$$d_{ij} = 2r \arcsin\left(\sqrt{\sin^2\left(\frac{\hat{\varphi}_{ij}-\varphi_{ij}}{2}\right) + \cos\hat{\varphi}_{ij}\cos\varphi_{ij}\sin^2\left(\frac{\hat{\lambda}_{ij}-\lambda_{ij}}{2}\right)}\right), \tag{48}$$

$$MPE = \frac{1}{n \times m}\sum_{i=1}^{n}\sum_{j=1}^{m} d_{ij}, \tag{49}$$

where d_{ij} denotes the Haversine distance between two coordinates. r is the radius of the earth sphere and $r = 6,371.01$ km. m stands for the coordinate number for each evaluated trajectory, and n denotes the sample size. For assessing the overall performance, the errors of all the predicted trajectories are averaged. The unit of MPE is in nautical mile (nm). For multiple trajectories evaluation, we use the averaged MPE.

C. Experimental settings

1) Parameter settings: For the iTransformer-based turning status prediction, Table I details the parameter settings. We set $N = 8$, embed dimension $d\_model = 16$, number of heads $n\_head = 2$, feed-forward layer dimension $d\_ff = 32$, and dropout rate $= 0.1$. For the training of GAN, some important parameters have been explicitly illustrated in Fig. 3 and Section III-C, and the setting for the remaining parameters can be referred to Table II. Meanwhile, Table III lists the parameter values for the trajectory prediction and enhancement modules. Parameters t_1, t_2 and t_3 are set as 10, 0.5 and 1 minute, respectively. Specifically, the turning status of vessel is predicted in real time with an interval of 0.5 minutes. Once a "turning" output is given, the trajectory prediction will be triggered every 1 minute, and it takes the past 10 minutes' trajectories as input. The prediction length of vessels'

trajectories is also 10 minutes as the interval of the AIS data is interpolated into 10 seconds and the sequence length L is set to 120.

TABLE I
PARAMETER SETTINGS FOR TURNING STATUS PREDICTION

Parameter	N	n_head	d_model	d_ff
Value	8	2	16	32
Parameter	DO	BS	TE	LR
Value	0.1	32	200	0.0001

Notes: "DO", "BS", "TE" and "LR" are short for "dropout rate", "batch size", "training epoch" and "learning rate", respectively.

TABLE II
PARAMETER SETTINGS FOR TRAJECTORY AUGMENTATION

Parameter	L	B	α	δ	TE	LR
Value	120	128	0.2	0.3	10^4	0.0002

Notes:"TE" and "LR" are short for "training epoch" and "learning rate", respectively.

TABLE III
PARAMETER SETTINGS FOR THE TRAJECTORY PREDICTION AND TRAJECTORY ENHANCEMENT

Parameter	t_1	t_2	t_3	S	k_1	k_2	l	ϵ
Value	10	0.5	1	6×10^5	100	5	6	0.2

TABLE IV
PERFORMANCE COMPARISON BETWEEN DIFFERENT MACHINE LEARNING METHODS

Model	Acc.	Pre.	Rec.	F1-score	AUC
SVM	0.9074	0.8793	0.9457	0.9113	*NA*
XGBoost	0.8899	0.8711	0.9167	0.8933	0.9553
SAE	0.8908	0.8755	0.9128	0.8937	0.9485
LSTM	0.8811	0.8582	0.9147	0.8856	0.9271
GRU	0.8889	0.8681	0.9186	0.8927	0.9418
TCN	0.9045	0.9019	0.9089	0.9054	0.9602
ATCN	0.9074	0.8949	0.9244	0.9094	0.9625
Transformer	0.6852	0.6199	**0.9671**	0.7555	0.8128
iTransformer	**0.9337**	**0.9211**	0.9496	**0.9351**	**0.9778**

Notes: NA means not applicable, respectively. The criterion of the AUC score is not applicable to the SVM model, because the standard SVM model is not able to output probability distributions.

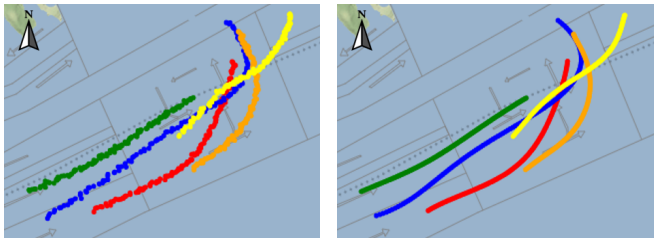
2) Other settings: The models in our study were developed using Python and TensorFlow, and we conducted all the experiments on a system equipped with an Intel(R) Xeon(R) Gold 6240 CPU @ 2.60GHz and NVIDIA GeForce RTX 2080 Ti GPU 11GB card. However, due to data confidentiality restrictions, we are unable to share the data sets online. If the readers need further information about the data sets, they may contact the authors directly.

D. Results about turning status prediction (answering RQ1)

The performance evaluation of the iTransformer model is shown in Table IV, where the iTransformer model exhibits impressive results, achieving high accuracy (93.37%), precision (92.11%), recall (94.96%), F1-score (93.51%), and AUC score (97.78%). For comparison, eight machine learning algorithms were chosen as baselines, including SVM, extreme gradient boosting (XGBoost), stacked autoencoder (SAE), LSTM, GRU, TCN, attention-based TCN (ATCN), and Transformer. The iTransformer model outperforms these in almost all metrics, except for precision. Both the ATCN and SVM models emerge as the top-performing baselines with an accuracy of 90.74%. The TCN model shows slightly lower results with 90.45% accuracy. The other models fall short of reaching 90% accuracy. The vanilla Transformer model ranks lowest, with only 68.52% accuracy. These results align with the findings in [55], suggesting that Transformers are less effective than linear models in time series forecasting.

E. Results about trajectory augmentation (answering RQ2)

Fig. 6 depicts five randomly generated trajectories from GAN, as well as these trajectories via fitting. It implies that the formulated trajectory fitting method is very capable of covering the trends of trajectory and generating a continuous and smooth trajectory.



(a) (b)

Fig. 6. Examples of trajectory fitting on five randomly generated trajectories. (a) the original trajectories generated by GAN. (b) the trajectories after fitting.

The forecasting error of several popular data augmentation methods is shown in Table V, along with the improvement percentage (IMP) of MSDL for each method. As a result of the shifting process, all the prediction errors decrease to less than 0.19 nm. MSDL introduces GAN for trajectory augmentation. The accuracy of predictions can be enhanced by employing GAN. In particular, only using the training data (i.e., real trajectories) without any data augmentation techniques results in an MPE of 0.1792 nm, whereas using GAN reduces it to 0.1644 nm, with an improvement of 8.26%. GAN is able to generate any number of additional synthetic trajectories with similar distributions of the training data. In this study, the scenario using GAN utilizes a total of 616,843 trajectories (i.e., 600,000 generated and 16,843 training trajectories). In contrast, the sample size for using training trajectories without any augmentation is only 16,843. Such a large discrepancy could be the reason for the significant enhancement in prediction accuracy. Furthermore, GAN is compared to other oversampling techniques that have been widely adopted for handling imbalanced data sets, including random oversampling, synthetic minority over-sampling technique (SMOTE), and adaptive synthetic (ADASYN). From the results demonstrated in Table V, we find that all three oversampling techniques help reduce prediction error. The best-performing oversampling method is SMOTE, with an error of 0.1722 nm. In contrast to the above three oversampling techniques, using GAN enhances

the performance by at least 4.53% (i.e., 0.1644 nm vs. 0.1722 nm). GAN is also compared with another generative model, i.e., the denoising diffusion probabilistic model (DDPM) [58]. Using GAN obtains better performance than using DDPM in vessel trajectory augmentation, with an improvement of 6.70% in this task.

TABLE V
PERFORMANCE COMPARISON BETWEEN OTHER SAMPLE ARGUMENTATION TECHNIQUES

Method	MPE (nm)	IMP (%)
Random Oversampling	0.2380	30.92
SMOTE	0.2285	28.05
ADASYN	0.2292	28.27
Training Trajectories	0.2609	36.99
Random Oversampling + shifting	0.1805	8.92
SMOTE + shifting	0.1722	**4.53**
ADASYN + shifting	0.1756	6.38
Training Trajectories + shifting	0.1792	8.26
DDPM + shifting	0.1762	6.70
MSDL	**0.1644**	—

Notes: "training trajectories" means purely using the training trajectories without any data augmentation.

F. Results about trajectory prediction (answering RQ3)

The primary consideration of proposing the trajectory prediction algorithm described in Section III-D is to represent the predicted trajectories with uncertainty. Unlike conventional machine learning techniques that typically generate a single trajectory, our algorithm has the capability to predict multiple potential trajectories, offering an advantage over standard machine learning approaches. To further confirm the efficiency and superiority of our proposed algorithm for VTP, we have also performed a comparison between our algorithm and using deep learning algorithms, including LSTM, bidirectional LSTM (Bi-LSTM), bidirectional GRU (Bi-GRU), attention-based Bi-LSTM and TCN. These methods train on the same data set as our algorithm does. The experimental performance has been shown in Table VI. The best-performing algorithm is Bi-LSTM, with an MPE of 0.1791 nm. However, it underperforms our algorithm (MSDL) by as high as 8.21% (0.1644 nm vs. 0.1791 nm).

TABLE VI
PERFORMANCE COMPARISON BETWEEN OUR METHOD AND USING MACHINE LEARNING METHODS

Method	MPE (nm)	IMP (%)
LSTM	0.3675	55.27
Bi-LSTM	0.1791	**8.21**
Bi-GRU	0.2056	20.04
Attention Bi-LSTM	0.1947	15.56
TCN	0.3949	58.37
MSDL	**0.1644**	—

TABLE VII
PERFORMANCE COMPARISON BETWEEN OUR METHOD WITH AND WITHOUT DYNAMIC CONTEXT-AWARE TRAJECTORY ENHANCEMENT

Method	MPE (nm)	IMP (%)
MSDL	0.1644	3.53
DCA-MSDL	**0.1586**	—

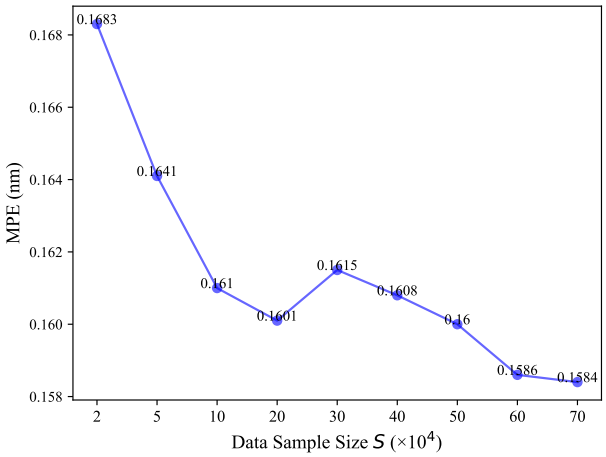


Fig. 7. Effect of size of the augmented training data sample for the DCA-MSDL model.

G. Results about dynamic context-aware trajectory enhancement (answering RQ4)

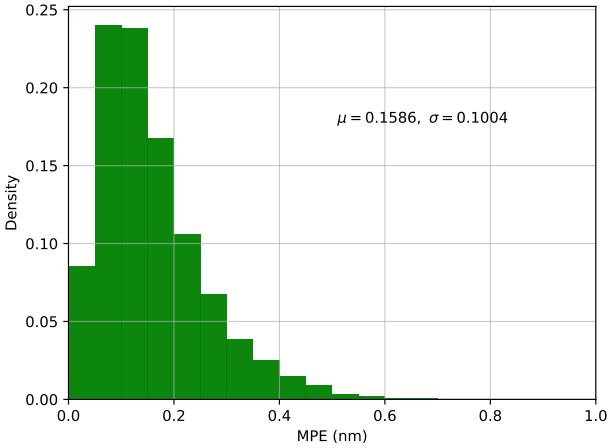
To validate the effect of the proposed dynamic context-aware trajectory enhancement in trajectory prediction, the DCA-MSDL algorithm is compared with the algorithm without trajectory enhancement (i.e., MSDL). As indicated in Table VII, DCA-MSDL further narrows the MPE from 0.1644 nm to 0.1586 nm, with a substantial improvement of 3.53%. Such an encouraging enhancement implies the critical impacts of the dynamic traffic context on vessels' trajectory prediction and the capability of our method for identifying such patterns.

Fig. 7 shows the performance of DCA-MSDL with different sizes of augmented data as the training data (S), from 2×10^4 to 6×10^5. Overall, the MPE decreases with more augmented data utilized. In particular, it drops by 5.88% (from 0.1683 nm to 0.1584 nm) with the data size increasing from 2×10^4 to 7×10^5, which further proves the efficiency of the trajectory augmentation technique from another viewpoint. Fig. 8 (a) plots the histogram of the prediction errors with our method. The mean and standard deviation values are 0.1586 nm and 0.1004 nm respectively. Fig. 8 (b) gives the percentage of errors in different ranges. For instance, 99.30% of errors are found within the range of $[0, 0.5]$, 90.76% are in the range of $[0, 0.3]$, and 73.17% are found within the range of $[0, 0.2]$. Additionally, Fig. 8 (c) shows the confidence interval (CI) at different levels, ranging from 95% to 75%. With a 95% confidence level, the positional error is specifically within the range of $[0, 0.3556]$. While the confidence interval (CI) narrows to $[0.0431, 0.2742]$ with a 75% confidence level.

H. Comparisons and analyses (answering RQ5)

In this section, our proposed method is comprehensively evaluated from two perspectives: one, the effect of different steps in the trajectory prediction algorithm, as described in Section III-D; and the other, the improvement of DCA-MSDL when compared to existing baseline methods in VTP.

The impacts of different procedures are compared and analyzed in Table VIII with regard to the following aspects:



(a) Histogram of Errors

range	[0, 0.1]	[0, 0.2]	[0, 0.3]	[0, 0.4]	[0, 0.5]
percentage	32.58%	73.17%	90.49%	96.88%	99.30%

(b) Error Range Percentage

confidence level	95%	90%	85%	80%	75%
CI	[0, 0.3555]	[0, 0.3238]	[0.0141, 0.3032]	[0.0299, 0.2874]	[0.0431, 0.2742]

(c) Confidence Interval (CI) Analysis

Fig. 8. Prediction positional error analyses. (a) histogram of the prediction errors. (b) percentage for certain error ranges. (c) confidence interval of the error at different confidence levels.

1) trajectory fitting vs. trajectory without fitting; 2) trajectory fitting vs. trajectory smoothing; 3) selection using the recent speed records vs. selection using all speed records; 4) using Euclidean distance vs. using DTW as the positional similarity. The experimental results exhibit that the step of shifting enhances forecasting accuracy considerably, with a dramatic improvement of 26.54%, lowering the MPE from 0.2159 nm to 0.1586 nm. The other procedures, such as fitting and recent speed selection, help improve the prediction accuracy by at least 0.44%. DTW helps enhance the accuracy by 1.37% than the Euclidean distance.

TABLE VIII
EFFECT OF DIFFERENT STEPS FOR THE TRAJECTORY PREDICTION ALGORITHM

Method	MPE (nm)	IMP (%)
fitting + shifting + DTW	0.1593	**0.44**
fitting + shifting + all speed selection + DTW	0.1660	4.46
smoothing + shifting + recent speed selection + DTW	0.1605	1.18
fitting + recent speed selection + DTW	0.2159	26.54
fitting + shifting + recent speed selection + Euclidean Distance	0.1608	1.37
DCA-MSDL (fitting + shifting + recent speed selection + DTW)	**0.1586**	—

Furthermore, nine existing baseline models are selected for performance comparison, including ELM [31], LSTM [34], GRU [54], Bi-LSTM [36], stacked Bi-GRU (SBi-GRU) [47], attention-based LSTM (ALSTM), transformer [51], LSTM seq2seq [40] and recurrent encoder-decoder networks (REDN) [48]. RNN-based frameworks and transformer have been popularly adopted for a variety of sequential data modeling tasks.

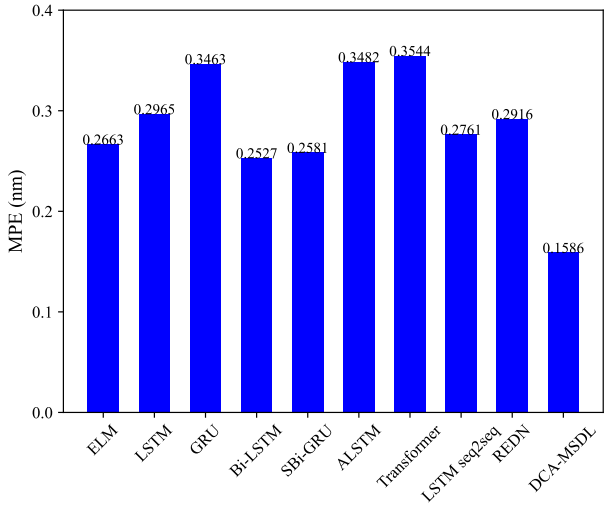
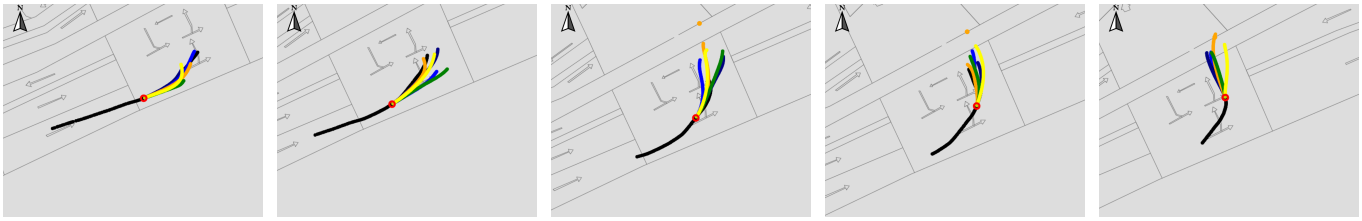


Fig. 9. Comparison of prediction error (i.e., MPE) between the DCA-MSDL and existing baseline methods.

One of the reasons is probably that they are very capable of learning long-term interdependencies [59]. To ensure a fair comparison, we also apply the methodology outlined in Section III-D-1) to fit all predicted trajectories produced by the baseline methods. Fig. 9 illustrates the prediction errors on the test data for DCA-MSDL as well as the above-mentioned baseline methods. All of the baselines' hyperparameters have been tuned carefully. Our constructed DCA-MSDL obtains the most accurate prediction with an MPE of only 0.1586 nm. Bi-LSTM, SBi-GRU, and ELM are the top three competitive baseline models, with MPEs of 0.2527 nm, 0.2581 nm, and 0.2663 nm, respectively. In the Bi-LSTM architecture, the sequential input data involves both forward and backward directions, which enables a higher learning capability on trajectory data. SBi-GRU deepens the whole network by stacking several bi-directional GRU layers to extract more abstract hidden features. The GRU converges faster than the LSTM and therefore improves the training efficiency. LSTM seq2seq ranks at the second place, with an MPE of 0.2761 nm. LSTM seq2seq incorporates frameworks of seq2seq structure and LSTM module, allowing more focus on specific parts of the input automatically. REDN and LSTM perform slightly poorer, with MPEs of 0.2916 nm and 0.2965 nm, respectively. GRU, ALSTM, and Transformer perform the worst, which implies that they are not suitable for this task. Compared to the existing approaches, the improvement of DCA-MSDL is significant, as it increases by at least 37.24% (i.e., 0.2527 nm to 0.1586 nm) in terms of prediction accuracy.

I. A case study illustration

We present a case study of a vessel in the Singapore Strait to explain how the DCA-MSDL works in order to gain a thorough understanding of our proposed DCA-MSDL approach. The vessel is entering the Port of Singapore with a destination of a pilot boarding ground (i.e., Eastern Boarding Ground "A" (PEBGA)). First, the turning status of this vessel



(a) 2018-01-01 01:49:10 (b) 2018-01-01 01:53:10 (c) 2018-01-01 01:57:10 (d) 2018-01-01 02:01:10 (e) 2018-01-01 02:05:10

Fig. 10. Illustration of DCA-MSDL for real-time VTP based on a real-world case study on 15 January 2028 in the Singapore Strait.

is predicted every 30 s by the trained iTransformer model. It is firstly predicted as “turning" status at timestamp 2018-01-01 01:49:10, where the vessel is located at the position labeled by a red hollow circle (○), as shown in Fig. 10 (a). Following that, a real-time trajectory prediction mechanism at an interval of 1 minute is triggered. For each prediction, the previous 10 minutes’ trajectories are extracted as the input of the model, then the model outputs five potential trajectories as the next 10 minutes’ trajectories. For simplicity, the predicted trajectories are shown at four-minute intervals from Fig. 10 (a) to (e), encoded by different colors including “navy" (●), “blue" (●), “orange" (●), “green" (●) and “yellow" (●). At timestamp 2018-01-01 02:05:10, the vessel has almost reached the traffic separation zone, which is a criterion for terminating the real-time prediction of trajectory; no further predictions are provided after this time.

V. Conclusions and Future Work

This paper proposes a dynamic context-aware approach based on multi-stage deep learning for vessel trajectory prediction. The core concept is first to use a deep learning framework based on iTransformer to predict a vessel’s turning status. Trajectory prediction is then conducted if the vessel is predicted as “turning". To assist in predicting vessel trajectory, a GAN framework is exploited for trajectory augmentation, and an algorithm based on multiple steps of nearest neighbor selection is contributed to trajectory prediction. Finally, trajectory enhancement is constructed to further improve prediction accuracy by incorporating the dynamic traffic context. Our approach is named DCA-MSDL (i.e., a dynamic context-aware approach based on multi-stage deep learning). The experiments on real-world data sets in Singapore show that the iTransformer framework for turning status prediction obtains a high accuracy at 93.37%. It also outperforms other classification models. The results also reveal the superiority of the established GAN framework in trajectory augmentation. Using the augmented samples generated by GAN decreases the prediction error by 8.26%. Furthermore, it significantly outperforms other commonly used oversampling techniques and the diffusion model. Comparison between the DCA-MSDL and existing methods for vessel trajectory prediction exhibits the advantages of DCA-MSDL, which achieves a prediction error as low as 0.1586 nm, with a dramatic improvement of at least 37.24%. The formulated trajectory enhancement mechanism by considering the dynamic context could significantly increase the accuracy by 3.53%. In future work, we will further improve our model by investigating other relevant features such as the ship type and length, because such conditions may affect the characteristics of a vessel’s movement pattern.

Acknowledgments

This work was supported in part by Maritime AI Research Programme (grant number SMI-2022-MTP-06 funded by Singapore Maritime Institute).

References

[1] P. Kaluza, A. Kölzsch, M. T. Gastner, and B. Blasius, “The complex network of global cargo ship movements,” *J. Royal Soc. Interface*, vol. 7, no. 48, pp. 1093–1103, 2010.

[2] E. Akyuz, “A hybrid accident analysis method to assess potential navigational contingencies: The case of ship grounding,” *Safety Sci.*, vol. 79, pp. 268–276, 2015.

[3] Z. Wan and J. Chen, “Human errors are behind most oil-tanker spills,” 2018.

[4] T. A. Johansen, T. Perez, and A. Cristofaro, “Ship collision avoidance and colregs compliance using simulation-based control behavior selection with predictive hazard assessment,” *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 12, pp. 3407–3422, 2016.

[5] M. Svanberg, V. Santén, A. Hörteborn, H. Holm, and C. Finnsgård, “Ais in maritime research,” *Marine Policy*, vol. 106, p. 103520, 2019.

[6] X. Zhang, X. Fu, Z. Xiao, H. Xu, and Z. Qin, “Vessel trajectory prediction in maritime transportation: current approaches and beyond,” *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 11, pp. 19 980–19 998, 2022.

[7] R. Zhao, D. Yan, Q. Liu, J. Leng, J. Wan, X. Chen, and X. Zhang, “Digital twin-driven cyber-physical system for autonomously controlling of micro punching system,” *IEEE Access*, vol. 7, pp. 9459–9469, 2019.

[8] D. Gao, Y. Zhu, J. Zhang, Y. He, K. Yan, and B. Yan, “A novel mp-lstm method for ship trajectory prediction based on ais data,” *Ocean Eng.*, vol. 228, p. 108956, 2021.

[9] Z. Xiao, X. Fu, L. Zhang, W. Zhang, R. W. Liu, Z. Liu, and R. S. M. Goh, “Big data driven vessel trajectory and navigating state prediction with adaptive learning, motion modeling and particle filtering techniques,” *IEEE Trans. Intell. Transp. Syst.*, 2020.

[10] L.-z. Sang, X.-p. Yan, A. Wall, J. Wang, and Z. Mao, “Cpa calculation method based on ais position prediction,” *J. Navig.*, vol. 69, no. 6, pp. 1409–1426, 2016.

[11] P. Last, M. Hering-Bertram, and L. Linsen, “Interactive history-based vessel movement prediction,” *IEEE Intell. Syst.*, vol. 34, no. 6, pp. 3–13, 2019.

[12] F. Mazzarella, V. F. Arguedas, and M. Vespe, “Knowledge-based vessel position prediction using historical ais data,” in *2015 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. IEEE, 2015, pp. 1–6.

[13] P. Last, C. Bahlke, M. Hering-Bertram, and L. Linsen, “Comprehensive analysis of automatic identification system (ais) data in regard to vessel movement prediction,” *J. Navig.*, vol. 67, no. 5, pp. 791–809, 2014.

[14] S. Hexeberg, A. L. Flåten, E. F. Brekke *et al.*, “Ais-based vessel trajectory prediction,” in *2017 20th international conference on information fusion (Fusion)*. IEEE, 2017, pp. 1–8.

[15] S. Hexeberg, "Ais-based vessel trajectory prediction for asv collision avoidance," Master's thesis, NTNU, 2017.

[16] D. Alizadeh, A. Alesheikh, and M. Sharif, "Vessel trajectory prediction using historical automatic identification system data," *J. Navigation*, vol. 74, no. 1, pp. 156–174, 2021.

[17] R. Scheepens, H. van de Wetering, and J. J. van Wijk, "Contour based visualization of vessel movement predictions," *Int. J. Geographical Inf. Sci.*, vol. 28, no. 5, pp. 891–909, 2014.

[18] M. Üney, L. M. Millefiori, and P. Braca, "Data driven vessel trajectory forecasting using stochastic generative models," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 8459–8463.

[19] L. M. Millefiori, P. Braca, K. Bryan, and P. Willett, "Modeling vessel kinematics using a stochastic mean-reverting process for long-term prediction," *IEEE Trans. Aerosp Electron. Syst.*, vol. 52, no. 5, pp. 2313–2330, 2016.

[20] C. Liu, S. Guo, Y. Feng, F. Hong, H. Huang, and Z. Guo, "L-vtp: Long-term vessel trajectory prediction based on multi-source data analysis," *Sensors*, vol. 19, no. 20, p. 4365, 2019.

[21] S. Guo, C. Liu, Z. Guo, Y. Feng, F. Hong, and H. Huang, "Trajectory prediction for ocean vessels base on k-order multivariate markov chain," in *13th International Conference on Wireless Algorithms, Systems, and Applications*. Springer, 2018, pp. 140–150.

[22] X. Zhang, G. Liu, C. Hu, and X. Ma, "Wavelet analysis based hidden markov model for large ship trajectory prediction," in *2019 Chinese Control Conference (CCC)*. IEEE, 2019, pp. 2913–2918.

[23] L. P. Perera, P. Oliveira, and C. G. Soares, "Maritime traffic monitoring based on vessel detection, tracking, state estimation, and trajectory prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1188–1200, 2012.

[24] Y. Lian, L. Yang, L. Lu, J. Sun, and Y. Lu, "Research on ship ais trajectory estimation based on particle filter algorithm," in *2019 11th International conference on intelligent human-machine systems and cybernetics (IHMSC)*, vol. 1. IEEE, 2019, pp. 305–308.

[25] B. Murray and L. P. Perera, "An ais-based multiple trajectory prediction approach for collision avoidance in future vessels," in *International Conference on Offshore Mechanics and Arctic Engineering*, vol. 58851. American Society of Mechanical Engineers, 2019, p. V07BT06A031.

[26] G. Pallotta, M. Vespe, and K. Bryan, "Vessel pattern knowledge discovery from ais data: A framework for anomaly detection and route prediction," *Entropy*, vol. 15, no. 6, pp. 2218–2245, 2013.

[27] A. L. Duca, C. Bacciu, and A. Marchetti, "A k-nearest neighbor classifier for ship route prediction," in *OCEANS 2017-Aberdeen*. IEEE, 2017, pp. 1–6.

[28] X. Liu, W. He, J. Xie, and X. Chu, "Predicting the trajectories of vessels using machine learning," in *2020 5th International conference on control, robotics and cybernetics (CRC)*. IEEE, 2020, pp. 66–70.

[29] J. Liu, G. Shi, and K. Zhu, "Vessel trajectory prediction model based on ais sensor data and adaptive chaos differential evolution support vector regression (acde-svr)," *Appl. Sci.*, vol. 9, no. 15, p. 2983, 2019.

[30] T. A. Volkova, Y. E. Balykina, and A. Bespalov, "Predicting ship trajectory based on neural networks using ais data," *J. Marine Sci. Eng.*, vol. 9, no. 3, p. 254, 2021.

[31] S. Mao, E. Tu, G. Zhang, L. Rachmawati, E. Rajabally, and G.-B. Huang, "An automatic identification system (ais) database for maritime trajectory prediction and data mining," in *Proceedings of ELM-2016*. Springer, 2018, pp. 241–257.

[32] E. Tu, G. Zhang, S. Mao, L. Rachmawati, and G.-B. Huang, "Modeling historical ais data for vessel path prediction: A comprehensive treatment," *arXiv preprint arXiv:2001.01592*, 2020.

[33] H. Li, H. Jiao, and Z. Yang, "Ais data-driven ship trajectory prediction modelling and analysis based on machine learning and deep learning methods," *Transp. Res. Part E Logist. Transp. Rev.*, vol. 175, p. 103152, 2023.

[34] H. Tang, Y. Yin, and H. Shen, "A model for vessel trajectory prediction based on long short-term memory neural network," *J. Marine Eng. Technol.*, pp. 1–10, 2019.

[35] S. Mehri, A. A. Alesheikh, and A. Basiri, "A contextual hybrid model for vessel movement prediction," *IEEE Access*, vol. 9, pp. 45 600–45 613, 2021.

[36] M. Gao, G. Shi, and S. Li, "Online prediction of ship behavior with automatic identification system sensor data using bidirectional long short-term memory recurrent neural network," *Sensors*, vol. 18, no. 12, p. 4211, 2018.

[37] C. Wang and Y. Fu, "Ship trajectory prediction based on attention in bidirectional recurrent neural networks," in *2020 5th International Conference on Information Science, Computer Technology and Transportation (ISCTT)*. IEEE, 2020, pp. 529–533.

[38] G. Liu and J. Guo, "Bidirectional lstm with attention mechanism and convolutional layer for text classification," *Neurocomputing*, vol. 337, pp. 325–338, 2019.

[39] M. Ding, W. Su, Y. Liu, J. Zhang, J. Li, and J. Wu, "A novel approach on vessel trajectory prediction based on variational lstm," in *2020 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*. IEEE, 2020, pp. 206–211.

[40] N. Forti, L. M. Millefiori, P. Braca, and P. Willett, "Prediction oof vessel trajectories from ais data via sequence-to-sequence recurrent neural networks," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 8936–8940.

[41] J. Sekhon and C. Fleming, "A spatially and temporally attentive joint trajectory prediction framework for modeling vessel intent," in *Learn. Dynamics Control*. PMLR, 2020, pp. 318–327.

[42] S. Capobianco, L. M. Millefiori, N. Forti, P. Braca, and P. Willett, "Deep learning methods for vessel trajectory prediction based on recurrent neural networks," *arXiv preprint arXiv:2101.02486*, 2021.

[43] P. Dijt and P. Mettes, "Trajectory prediction network for future anticipation of ships," in *Proceedings of the 2020 International Conference on Multimedia Retrieval*, 2020, pp. 73–81.

[44] J. Venskus, P. Treigys, and J. Markevičiūtė, "Unsupervised marine vessel trajectory prediction using lstm network and wild bootstrapping techniques," *Nonlinear Anal. Model. Control*, vol. 26, no. 4, pp. 718–737, 2021.

[45] L. You, S. Xiao, Q. Peng, C. Claramunt, X. Han, Z. Guan, and J. Zhang, "St-seq2seq: A spatio-temporal feature-optimized seq2seq model for short-term vessel trajectory prediction," *IEEE Access*, vol. 8, pp. 218 565–218 574, 2020.

[46] Y. Xiao, X. Li, W. Yao, J. Chen, and Y. Hu, "Bidirectional data-driven trajectory prediction for intelligent maritime traffic," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 1773–1785, 2023.

[47] Y. Xu, J. Zhang, Y. Ren, Y. Zeng, J. Yuan, Z. Liu, L. Wang, and D. Ou, "Improved vessel trajectory prediction model based on stacked-bigrus," *Secur. Commun. Netw.*, vol. 2022, 2022.

[48] S. Capobianco, N. Forti, L. M. Millefiori, P. Braca, and P. Willett, "Recurrent encoder-decoder networks for vessel trajectory prediction with uncertainty estimation," *IEEE Trans. Aerosp. Electron. Syst.*, 2022.

[49] X. Chen, J. Ling, Y. Yang, H. Zheng, P. Xiong, O. Postolache, and Y. Xiong, "Ship trajectory reconstruction from ais sensory data via data quality control and prediction," *Math. Probl. Eng.*, vol. 2020, 2020.

[50] K.-I. Kim and K. M. Lee, "Deep learning-based caution area traffic prediction with automatic identification system sensor data," *Sensors*, vol. 18, no. 9, p. 3172, 2018.

[51] D. Nguyen and R. Fablet, "Traisformer-a generative transformer for ais trajectory prediction," *arXiv preprint arXiv:2109.03958*, 2021.

[52] B. Murray and L. P. Perera, "A data-driven approach to vessel trajectory prediction for safe autonomous ship operations," in *2018 Thirteenth International Conference on Digital Information Management (ICDIM)*. IEEE, 2018, pp. 240–247.

[53] B. Murray and L. Perera, "A dual linear autoencoder approach for vessel trajectory prediction using historical ais data," *Ocean Eng.*, vol. 209, p. 107478, 2020.

[54] Y. Suo, W. Chen, C. Claramunt, and S. Yang, "A ship trajectory prediction framework based on a recurrent neural network," *Sensors*, vol. 20, no. 18, p. 5133, 2020.

[55] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long, "itransformer: Inverted transformers are effective for time series forecasting," *arXiv preprint arXiv:2310.06625*, 2023.

[56] N. Sriram, B. Liu, F. Pittaluga, and M. Chandraker, "Smart: Simultaneous multi-agent recurrent trajectory prediction," in *Eur. Conf. Comput. Vision*. Springer, 2020, pp. 463–479.

[57] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Adv. Neural Inf. Process. Syst.*, vol. 27, 2014.

[58] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 6840–6851, 2020.

[59] X. Zhang, Z. Zhao, and J. Li, "Arde-n-beats: An evolutionary deep learning framework for urban traffic flow prediction," *IEEE Internet Things J.*, vol. 10, no. 3, pp. 2391–2403, 2023.

Xiaocai Zhang received a Ph.D. degree in computer science from the University of Technology Sydney, Australia, in 2021, and a B.S. degree in navigation technology and an M.S. degree in traffic & transportation engineering from Dalian Maritime University, Dalian, China, in 2013 and 2016, respectively. He is currently a Research Scientist with the Institute of High Performance Computing, A*STAR, Singapore. He was a Research Assistant with the Data Science Institute, University of Technology Sydney, Australia. His research interests include data mining, deep learning, urban computing, time-series analysis and intelligent transportation systems.

Xiuju Fu received the B.S. and M.S. degrees from the Beijing Institute of Technology, China, in 1995 and 1999, respectively, and the Ph.D. degree from Nanyang Technological University, Singapore, in 2003. She is currently a Senior Research Scientist and group manager with the Institute of High Performance Computing, A*STAR. She has been a Principal Investigator (PI) and co-PI of several grant projects. She has co-authored dozens of papers in conference proceedings and journals. Her research interests include complex network analytics, modeling and simulation, and data mining. In recent years, she is active in developing big data intelligence and modeling and simulation approaches for applications in maritime traffic safety and operation efficiency, supply chain and public health areas.

Zhe Xiao received the Ph.D. degree in Nanyang Technological University, Singapore, in 2015. He joined the Institute of High Performance Computing, A*STAR as a research scientist and worked in data science, agent-based modeling, big data analytics and participatory sensing applications. He closely worked with industry to develop real-world computing applications. His research interests include system architecture design, data mining, AI and machine learning, data privacy protection and large-scale concurrent system design and implementation.

Haiyan Xu is a senior research scientist at the Institute of High Performance Computing, A*STAR, Singapore. She received her M.S. (2002) in Applied Mathematics and Ph.D. (2005) in Computational Statistics. She has published more than 30 papers in peer-reviewed conferences and journals such as IEEE Transactions on Reliability, PLOS Neglected Tropical Diseases, Computational Statistics & Data Analysis, The International Journal of Advanced Manufacturing Technology, PLOS One, Physica A: Statistical Mechanics and its Applications, etc. Her research interest is statistical learning, including longitudinal data analysis, growth mixture model, clustering analysis, predictive model, risk evaluation, accelerate life/degradation testing design and data analysis, etc.

Wanbing Zhang received his B.E. degree in optoelectronics technology from Beijing Institute of Technology. He is a senior research engineer with the Institute of High Performance Computing, A*STAR, 138632, Singapore. His research interests include software architecture design, simulation approaches, large-scale distributed computing, and data analysis. He had focused on software design and architecture more than ten years and had been involved in several large systems as key person.

Jimmy Koh heads the Pilotage Services Department of PSA Marine. Leading his team of managers and 280 harbour pilots, they deliver pilotage services to vessels that call at the Port of Singapore safely, professionally, reliably, and efficiently. Jimmy also spearheads digital transformation to streamline pilotage work processes and enhance situational awareness. He has been key in developing ONE-HANDSHAKE™ - an integrated digital platform that supports connectivity across maritime stakeholders and provide real-time communication and alerts to the shipping community. Since 2020, Jimmy was appointed as a member of the National Maritime Safety at Sea Council. He was also appointed a member of working group on shore-based mooring and unmooring by the TC on Personal Safety and Health of the Singapore Manufacturing Federation.

Zheng Qin received the Ph.D. degree from National University of Singapore, in 2006, and the B.Eng. degree from Xi'an JiaoTong University, in 2001. He is a Senior Scientist and the Director of the Systems Science Department at the Institute of High Performance Computing, A*STAR. The department has strong capabilities in complex systems, multi-scale optimisation, and energy systems. His research interests include cloud computing, transport and logistics, and complexity science. He was Program Committee Chair of ICPADS 2018 and Workshop Co-Chair of ICDCS 2020. He serves on the Editorial Board of IEEE Transactions on Parallel and Distributed Systems.