Ocean Engineering 312 (2024) 119186

Contents lists available at ScienceDirect

Ocean Engineering

journal homepage: www.elsevier.com/locate/oceaneng

Research paper

SSRLM: A self-supervised representation learning method for identifying one ship with multi-MMSI codes

Yifeng Sun [a,b], Yaochi Zhao [a,*], Zhuhua Hu [a,b], Wei Wu [a], Jingwen Xia [a], Yizhen Wang [c]

[a] School of Information and Communication Engineering, School of Cyberspace Security, Hainan University, Haikou, 570228, China
[b] State Key Laboratory of Marine Resource Utilization in South China Sea, Hainan University, Haikou, 570228, China
[c] College of Physics and Electronic Engineering, Hainan Normal University, Haikou, 571158, China

ARTICLE INFO

Keywords:
Maritime Mobile Service Identity(MMSI)
One Ship with Multiple MMSI Codes
Self-supervised representation learning
Pre-training
Fine-tuning

ABSTRACT

It is of great practical significance for maritime safety and shipping development to achieve "One Ship with One Maritime Mobile Service Identity (MMSI) Code" in accordance with regulations. However, the current "One Ship with Multiple MMSI Codes" violation identification method lacks practicability and generalization. Aiming at the these problems, this paper proposes a self-supervised representation learning model for the "One Ship with Multiple MMSI Codes" abnormal behavior recognition task, named SSRLM. Firstly, we construct a new dataset about "One Ship with Multiple MMSI Codes", named HN_MulMI. Secondly, the SSRLM model learns the contextual interdependencies among different trajectory features in the HN_MulMI dataset through an unsupervised pre-training scheme, and extracts a dense vectorial representation of the ship's motion trajectories. Finally, by using the unique characteristics of the "One Ship with Multiple MMSI Codes" trajectory sequence, the SSRLM model captures its feature dependencies through a fine-tuning scheme to accomplish the abnormal behavior detection task. Experimental results in the two scenarios of HN_MulMI test set demonstrate that the proposed model outperform the state-of-the-art models in recent years, the average precision, recall and F1-score rates is up to about 99.26% and 93.5%. Meanwhile, we also conducted comparative experiments on three public datasets to verify the generalization performance, achieving 90.31%, 93.2% and 95.14% results in the F1 score evaluation indicators, further confirming the good recognition performance of the SSLLM model. Finally, our model has been applied to the actual scene and achieved good results.

1. Introduction

As an essential navigational tool for avoiding collisions and exchanging information between ships, an increasing number of ships are equipped with Automatic Identification System (AIS) equipment, which provides effective ship situational awareness at sea (Vesecky et al., 2009). Through AIS equipment, the ship can send static information such as ship name, identification code, destination, whether to load dangerous goods, etc. to the AIS base station (Zhang et al., 2016). Port and shipping management and other related departments also can use AIS equipment to obtain the ship's sailing plan, actual position, etc. Wu et al. (2018), so as to plan port traffic. AIS has strengthened and enhanced the management functions of ship traffic services Eriksen et al. (2006).

According to the regulations, each ship is required to install only one AIS device, and each AIS device has only one unique MMSI code (Xiao et al., 2015), which is used to facilitate the identification of ships and to improve the efficiency of maritime search and rescue. No unit or individual shall use MMSI code in violation of the regulations, which will affect the maritime search and identification. "One Ship with Multiple MMSI Codes" violations mainly include arbitrary tampering with the MMSI code, installing and turning on multiple AIS devices with inconsistent MMSI codes (Mazzarella et al., 2016; Iphar et al., 2020). Fig. 1 depicts two typical trajectories of "One Ship with Multiple MMSI Codes" behavior, the red and yellow curves indicate the traces of two different MMSI codes acquired, respectively. Specifically, the typical characteristics of tampering with MMSI code are that the MMSI code changes in the same ship trajectory; The typical feature of installing and turning on multiple AIS devices with different MMSI codes is that the tracks of multiple MMSI codes overlap to a high degree. Tampering with MMSI codes may trigger identity confusion, making it difficult for regulatory and law enforcement authorities to accurately trajectory and manage ships. The act of installing and turning on multiple AIS devices with inconsistent MMSI codes may trigger communication confusion and interfere with normal ship communication and navigation systems.

* Corresponding author.
E-mail address: zhyc@hainanu.edu.cn (Y. Zhao).

https://doi.org/10.1016/j.oceaneng.2024.119186
Received 8 January 2024; Received in revised form 31 August 2024; Accepted 3 September 2024
Available online 12 September 2024

(a) The trajectory of turning on multiple AIS devices with inconsistent MMSI codes (b) The trajectory of tampering with MMSI codes

Fig. 1. Chart of two typical trajectories characterizing the "One Ship with Multiple MMSI Codes" behavior.

Therefore, it is essential to effectively identify ships exhibiting the phenomenon of "One Ship with Multiple MMSI Codes".

In general, the mainstream research directions in the field of abnormal ship behavior detection are divided into two types: knowledge-driven methods and data-driven methods (Wang et al., 2023). The knowledge-driven methods rely on expertise and experience in the maritime field to assess whether a ship's behavior is abnormal or not. However, due to the influence of various external objective environments, the exact definition of abnormal behavior often appears to be ambiguous, and it is difficult to clearly classify whether a ship belongs to abnormality by using specific indicators. The existence of these problems reduces the practicality of the knowledge-driven ship abnormal behavior detection methods.

The data-driven methods help to deeply explore the ship behavioral characteristics, activity patterns, and feature correlations between various sensor data. With the advancement of Artificial Intelligence(AI), Deep Learning(DL) methods have provided new directions for data-driven research in detection of abnormal ship behavior (Murray and Perera, 2020). DL methods are typically categorized into two types based on dataset types: supervised learning methods and unsupervised learning methods. Supervised learning methods can achieve good results with sufficient sample data and labels. However, in the field of detection of abnormal ship behavior, the abnormal trajectory samples are scarce and difficult to label, so it is difficult to carry out the supervised learning ship abnormal behavior detection method based on classification.

Although unsupervised learning can overcome the challenges of labeling, there are still some unresolved problems. Firstly, due to limitations in the sensors themselves, there are noise, errors and missing values in ship trajectory data, and how unsupervised learning can learn the temporal dependence and spatial dependence of ship trajectories more robustly under this premise. Secondly, due to the diversity of potential abnormal behavior types, whether unsupervised learning is applicable to the field of ship abnormal behavior recognition and whether it increases the unpredictability of the training data also needs further research.

In order to solve these problems, we proposed a "One Ship with Multiple MMSI Codes" anomalous behavior recognition method based on the self-supervised representation learning, named SSRLM.

The main contributions of this paper are summarized as follows:

- We produce a new "One Ship with Multiple MMSI Codes" anomalous behavior identification dataset, named HN_MulMI, which collects and fuses AIS and radar data from ships around the offshore area centered in Hainan Province, China.
- We propose a "One Ship with Multiple MMSI Codes" recognition model based on a two-stage training strategy, named SSRLM. Its implementation is divided into the pre-training phase and the fine-tuning phase. SSRLM adopts the pre-training method of random mask to shield some trajectory features and reconstruct these features. Meanwhile, the length and proportion of the mask can be artificially controlled, enabling the model fully considers the overall distribution of input trajectory and alleviates the distribution deviation problem.
- In order to solve the problem of difficulty in obtaining sufficient labeled data for training, the SSRLM model chooses to use a large amount of unlabeled trajectory data in the pre-training stage, and the trajectory samples can generate different views each time they enter the model training due to random mask, so that the model learns the motion characteristics of the entire ship trajectory data.
- To address the problem of unpredictability caused by multiple potential anomalies in the trajectory data, SSRLM can distinguish the "One Ship with Multiple MMSI Codes" type trajectory by using only a small amount of label data to fine-tune the model on the basis of pre-training model, which greatly saves the high time cost of manually labeling datasets.
- In the experimental phase, the real ship trajectory dataset HN_MulMI from Hainan Province of China is used for training and testing, and the contribution of each component is verified through experiments. Meanwhile, experiments are conducted on the public abnormal behavior detection dataset to demonstrate the excellent ability of SSRLM model to recognize abnormal behavior in the field of abnormal behavior detection. Finally, our model has been deployed on real-world offshore management platform and has achieved excellent performance.

The rest of the paper is organized as follows: Section 2 describes the work related to ship abnormal behavior detection method, the Transformer-based time series models and pre-trained models. Section 3 provides the SSRLM model of this work in detail. Section 4 presents the experimental specific details and results. Finally, Section 5 summarizes the contributions of this work and the outlook for future directions.

2. Related work

2.1. Ship anomaly behavior detection methods

To facilitate the organization of the research progress in ship anomaly behavior detection, as illustrated in Fig. 2, we have compiled a RoadMap outlining the studies related to ship anomaly behavior detection methods.

Knowledge-driven methods generally rely on specialized empirical knowledge related to the maritime domain to assess whether a ship's behavior is abnormal. Kazemi et al. (2013) proposes a decision support system based on open data and expert rules that identifies 11 rules for abnormal ship behavior through consultation with several domain experts. Terroso-Saenz et al. (2016) proposed a method that follows the Complex Event Processing (CEP) paradigm to classify ship abnormal behavior into single-vessel abnormal behavior and multi-vessel abnormal behavior, providing a finer classification. Chen et al. (2014) proposed a knowledge discovery system based on genetic algorithms for developing rules for discriminating the risk level of ship behavior. Soares et al. (2019) proposes a framework that supports the introduction of external rule sets for the identification of ship behaviors that do not conform to the rules.

However, in practical applications, discernment methods based on knowledge typically involve setting rules and thresholds. Due to the influence of various factors, the precise definition of ship anomaly behavior often appears ambiguous, and it is difficult to make it clear with the rules. Furthermore, the reasonableness of the thresholds directly determines the accuracy of the abnormality identification, but it is difficult to use specific thresholds to classify whether a ship is abnormal according to various situations. Moreover, methods for converting knowledge and experience about ship anomalies into a

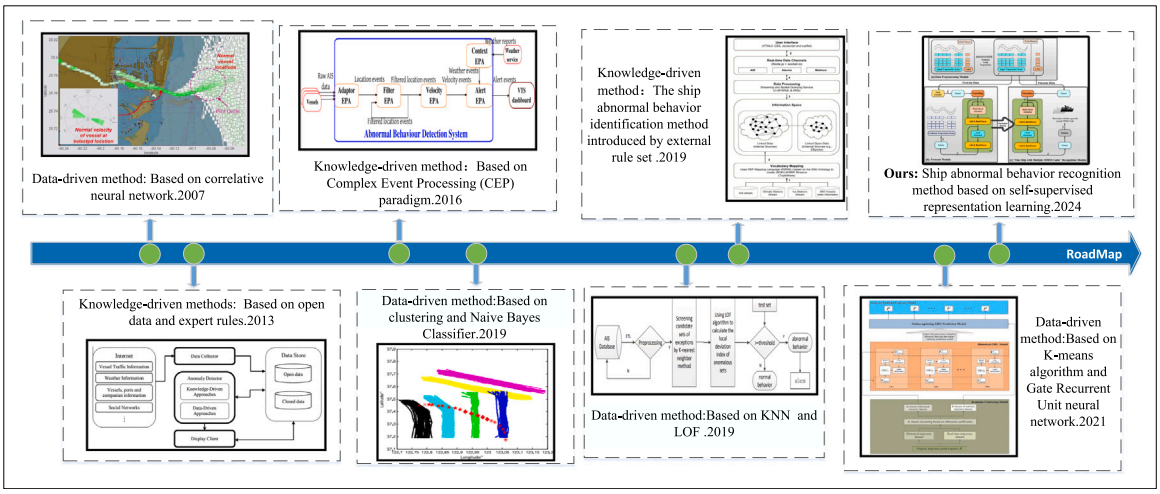


Fig. 2. Development of ship anomaly behavior detection.

knowledge base are often effective for judging relatively simple anomalous behavior, and are often ineffective in discriminating anomalous behaviors that require joint consideration of multiple factors. The existence of these problems reduces the practicality and generalizability of knowledge-driven ship abnormal behavior detection methods.

Data-driven methods for ship anomaly behavior detection involve to analyze large-scale data and mine the laws and trends hidden behind the data. With the increase of ship information data modality, the AIS equipment can receive more ship data information collected by sensors, which provides more research directions for the research of data-driven ship abnormal behavior detection. Rong et al. (2020) based on the static information of ships, groups AIS data and then uses the lateral distribution of trajectories and speeds to detect whether a ship's route and heading are abnormal. Zhen et al. (2017) clusters ship trajectories and then fuses the clustering results with a Naive Bayes classifier for the detection of abnormal ship behavior. Huang and Zhang (2019) employs a detection method based on global and local variables, combining the K-Nearest Neighbor (K-NN) algorithm with the Local Outlier Factor (LOF) algorithm to perform the task of detecting ship anomaly trajectories. Rhodes et al. (2007) utilizes an unsupervised incremental learning approach based on an Association Neural Networks to predict ship behavior. Subsequently, by comparing the prediction results with the actual truth, the trajectory's abnormality is determined.

However, such data-driven ship abnormal behavior detection studies based on statistical and clustering methods are generally based on screening models constructed on the basis of historical data in a particular water area (Gu et al., 2024), and the models usually need to be reconstructed for abnormal AIS data in other water areas, and this abnormal data screening method also lacks an effective data reliability model for data assessment.

With the development of AI, the research on DL-based methods for detecting abnormal ship behavior has been carried out successively. Zhao and Shi (2019) performs DBSCAN clustering on ship trajectories, and uses the clustering results as input to the Long-Short Term Memory (LSTM) network to construct a ship anomaly trajectory recognizer. Yin et al. (2020) used a two-stage sliding window approach to extract data features, and combined Convolutional Neural Networks (CNN) with Recursive Autoencoders to improve the effectiveness of detecting abnormal behavior in time series. Singh and Heymann (2020) proposed an anomaly detection framework based on multi class Artificial Neural Networks to classify intentional and unintentional AIS switch anomalies. Han et al. (2021) introduced an online learning trajectory prediction method that combines the K-means algorithm and Gate Recurrent Unit (GRU) neural network for ship abnormal behavior discrimination.

However, these above research directions of abnormal behavior detection of ships are still mainly focused on the problem of abnormal single-dimensional features such as speed, heading, position, etc., and these single-dimensional features tend to have a large difference compared with the normal values; and less attention is paid to the detection of abnormal behaviors that need to be considered jointly with multi-dimensional features in real life, and the abnormal behaviors that need to be considered jointly with multi-dimensional features in real life tend to be normal in a single dimension, but combined together will form abnormal behavior. Moreover, some of the above methods directly judge the trajectories with low probability of occurrence as abnormal trajectories, and it is difficult to objectively and accurately assess test results using these methods.

At present, there are fewer studies on the detection of "One Ship with Multiple MMSI Codes" violations. In real life, the detection of "One Ship with Multiple MMSI Codes" illegal behavior is still mainly based on manual on-site inspection in ports and other locations to check whether the ship's AIS equipment has been tampered with or more than one AIS equipment has been installed (Guan et al., 2023). Zhang et al. (2020) adopts the method based on D-TRAP and Correlation Map to detect the behavior of AIS shutdown and MMSI tampering, and some of the ideas can be used as a reference for the detection of "One Ship with Multiple MMSI Codes" behavior. Guan et al. (2023) employs a method that combines interval sampling and cosine similarity to calculate ship trajectory similarity, considering trajectories as "One Ship with Multiple MMSI Codes" if the similarity exceeds a certain threshold. However, these checking methods have the same drawbacks as the above statistical methods.

In summary, there are relatively few studies on the practical problem of "One Ship with Multiple MMSI Codes" detection. The existing methods fail to consider the actual navigation conditions and the influence of navigation environments in different regions. In different waters, the behavior characteristics of ships in the waters need to be reconsidered, and it takes a long time to reconstruct the model, which is inefficient. Therefore, this study proposes an accurate, efficient and general "One Ship with Multiple MMSI Codes" trajectory recognition model based on the DL domain.

2.2. *Transformer-based time series model and pretrain model*

The outstanding ability of the Transformer model to capture long-term dependencies and interactions with each other is particularly attractive for time series modeling, achieving excellent results in many time series applications, and variants of the model can also be used in a variety of domains to deal with a wide range of specific problems (Vaswani et al., 2017). TranAD proposed an adversarial training

procedure to amplify reconstruction errors considering that Transformer-based networks tend to miss smaller anomalous deviations in the time series (Tuli et al., 2022). AnomalyTrans proposed a Minimax Strategy in conjunction with the Transformer network structure to amplify the extent to which rare anomalies are distinguishable from normal time series data (Xu et al., 2021). Transformer network is particularly suitable for processing time series data, its multi-head attention can simultaneously focus on information from different locations in different representation subspaces, with the ability to mine the feature relationships between the input data. However, the transformer structure also suffers from high time complexity and drawbacks inherent to its structure, such as the encoder–decoder model that suffers from vanishing long-range gradients.

Pre-trained models are neural network learning methods that are pre-trained on a dataset and are utilized to initialize the model parameters and then migrate the trained model on top of a specific task. Pre-training methods for mask reconstruction are widely used in Computer Vision (CV) domain. In CV domain, MAE chose to mask most of the patches directly during pre-training, keeping only a small portion of the visible patches received by the encoder (He et al., 2022). LoMaR uses local mask reconstruction by performing mask reconstruction within patches of a small window on the Transformer encoder (Chen et al., 2022). The effects of the (Hou et al., 2022; Feichtenhofer et al., 2022) have also shown that MAE-style pre-training methods are effective for representation learning. In the Natural Language Processing (NLP) domain, BERT is the first unsupervised, deeply bi-directional pre-training system for natural language processing models, and this bi-directionality helps the models to better understand the context of the text (Devlin et al., 2018). In addition, BERT can perform different natural language processing tasks simultaneously. Zerveas et al. (2021) adopts the strategy of masking a small portion of the time series for characterization learning. TSformer similarly used masked post-reconstruction for the pre-training task on time series data (Shao et al., 2022).

The SSRLM model proposed in this paper inherits the advantages of Transformer in modeling time series dependencies. It utilizes an improved Transformer encoder structure to learn trajectory sequence data. This asymmetric structure effectively reduces the training parameter volume, saves computational resources, and mitigates the risk of overfitting to some extent. It also alleviates the problem of long-range gradient vanishing. Meanwhile, the SSRLM model draws on the success of MAE style and BERT style in pre-training, and adopts the pre-training method of randomly masking the features and then predicting them through the contextual content, so that the model better learns the contextual content of the trajectory sequence, and adequately captures the motion characteristics of the ship's trajectory sequence and the potential supervisory information.

3. Proposed method

3.1. Problem definition

This subsection provides a formal definition of the anomalous ship trajectory identification problem. The ship trajectory dataset can be regarded as a collection of trajectory points, and the dataset is formally represented as shown in Eq. (1):

$$\begin{cases} D = \{X_1, X_2, \ldots, X_n\} \\ X_i = \{x_{i1}, x_{i2}, \ldots, x_{iT}\} \\ x_{ij} = (\text{ Latitude }_{ij}, \text{ Longitude }_{ij}, \text{ Speed }_{ij} \ldots) \end{cases} \tag{1}$$

where the dataset $D = \{X_1, X_2, \ldots, X_n\}, n = 1, 2, 3, \ldots$, is a collection of multiple ship trajectory sequences , with each X representing a complete trajectory sequence. For a set of ship trajectory sequences $X_i = \{x_{i1}, x_{i2}, \ldots, x_{iT}\} \in R^{T*m}, i = 1, 2, \ldots, n$, with length T in the ship trajectory dataset D, the trajectory points sampled at time jth are represented as $x_{ij} = (\text{ Latitude }_{ij}, \text{ Longitude }_{ij}, \text{ Speed }_{ij} \ldots), j = 1, 2, \ldots, T$, where Latitude_{ij}, Longitude_{ij} and Speed_{ij} denote the latitude, longitude, and speed of the trajectory sequence X_i at the jth moment. In addition to that, x_{ij} also contains other ship trajectory features of X_i at the jth moment, such as timestamp, heading, course, etc.

For the pre-training task of the SSRLM model, the masked feature attribute value at the time point t is predicted according to the feature attributes of the preceding c trajectory points $(x_{t-c}, x_{t-c+1}, \ldots, x_{t-1}) \in R^{c*m}$ and the following d trajectory points $(x_{t+1}, x_{t+2}, \ldots, x_{t+d}) \in R^{d*m}$ of the trajectory sequence X at the time point t, and the prediction result is defined as $\tilde{x}_t$.

For the task of detecting abnormal behavior of "One Ship with Multiple MMSI Codes", the basic truth matching classification is performed based on each trajectory sequence X and its corresponding label y. According to the data of each trajectory sequence X, the model learns to predict the distribution $\hat{y}$ of the trajectory on the class, and to judge whether the trajectory sequence belongs to the behavior of "One Ship with Multiple MMSI Codes", and then compare the predicted labels $\hat{y}$ with the real labels y, to judge whether the recognition is correct or not.

3.2. Model architecture

The overall structure of the SSRLM is illustrated in Fig. 3, which can be divided into three modules. Module [a] is the data preprocessing module, which mainly performs cutting operations on excessively long trajectory data so that the model can learn better. The input and output of this module are two dimensional matrices representing the trajectory sequence; Module [b] is the pretrain module, which is the main part of the network model. It is primarily composed of the Transformer encoder with multiple layers of multi-head self-attention mechanisms. The input of this module is the trajectory sequence X mapped to two-dimensional matrix, and the output is the reconstructed trajectory sequence $\tilde{X}$ after masking X; Module [c] is the "One Ship with Multiple MMSI Codes" recognition module, which is used to evaluate whether the input trajectory sequence X exhibits abnormal behavior related to "One Ship with Multiple MMSI Codes". The input of this module is the trajectory sequence X mapped to two-dimensional matrix, and the output is the prediction label $\hat{y}$ corresponding to X.

To address the problem of difficulty in obtaining sufficient labeled data in deep learning, the pretrain module designs a denoised autoregressive task to extract a dense vector representation of ship trajectory sequences. Subsequently, aiming at the situation of multiple potential abnormal trajectories in trajectory data, SSRLM can solve the problem of possible unpredictability by downstream fine-tuning using only a small amount of labeled data. And due to the characteristics of DL, even when transitioning to different water areas, it only requires introducing the trajectory data of that water area and retraining with an expanded dataset.

The following subsections provide detailed explanations of each module in the SSRLM model.

3.2.1. Data preprocessing module

After performing steps such as feature selection and resampling, the HN_MulMI dataset exhibits significant differences in the lengths of different trajectory sequences. Some trajectory lengths consist of several thousand rows, while some trajectories have lengths of around 20 rows. In this case, the model will be more inclined to deal with the short trajectory sequences that are more common in the HN_MulMI dataset, and will not perform well for the small number of long trajectory sequences. This could result in a decrease in the overall model accuracy. Therefore, we choose to truncate long sequences before they enter the SSRLM model. Sequences with lengths exceeding 300, representing trajectories lasting at least 5 h, are truncated once or multiple times. Each truncation ensures that the length of the truncated subtrajectory

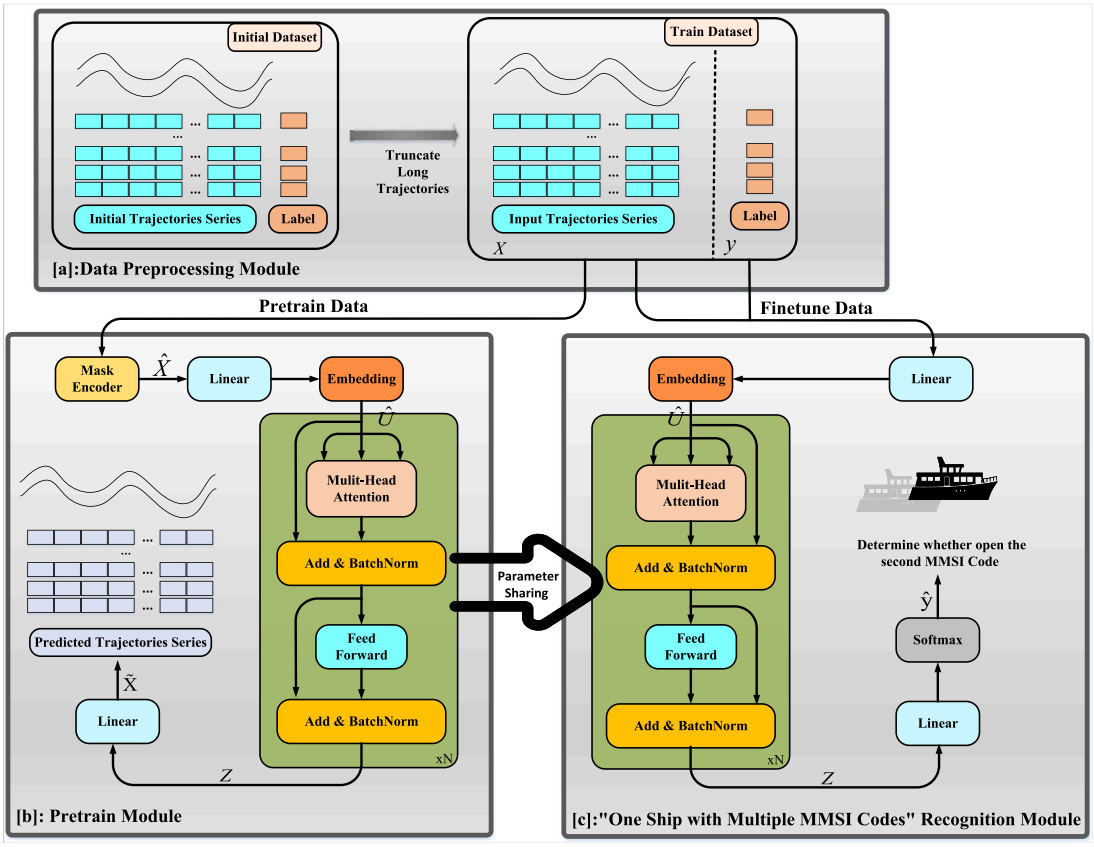


Fig. 3. Overall architecture of the SSRLM model.

remains within 300, and new IDs and labels are automatically assigned to the truncated subtrajectory sequences. Subsequently, the trajectory sequences undergo standardization before being input into the model network.

3.2.2. Pretrain module

The main component of the pretrain module is the Transformer encoder block consisting of multiple layers with the multi-head self-attention mechanism. The masked trajectory sequence $\hat{X}$ is mapped to a low-dimensional hidden encoding vector representation, which captures the key information of the input data, and then the original trajectory data X is restored from this hidden coded vector representation to minimize the reconstruction error.

The architecture used in the pretrain module is illustrated in Fig. 4, where feature information is extracted from ship trajectory data by denoising autoregressive method. The specific mask implementation is to create a binary noise mask $M \in R^{T*m}$ for each training sample and epoch independently, and then randomly mask the trajectory data features by element multiplication, allowing the model to predict the masked values, and the whole mask reconstruction process learns directly from the trajectory sequence data without the help of the labels and any a priori information during the process. And this strategy of random mask reconstruction allows the same sample to produce different views into the encoder at each training, fully learning the motion characteristics of the whole ship trajectory.

3.2.2.1. Improved transformer encoder. Generally in NLP tasks, the Transformer model uses Layer Normalization because word embeddings in NLP tasks are not affected by outliers. However, due to hardware sensor performance issues, ship trajectory data often contain outliers or noise points when received, which makes the Layer Normalization method more affected in calculating the mean and variance of separate samples in the presence of anomalous data, whereas the Batch Normalization method applies each computation to a single feature in a batch of data, which reduces the sensitivity of the anomalous data to the training of the network. Meanwhile, the main disadvantage of Batch Normalization is that it is difficult to adapt to samples of widely varying lengths, whereas our samples are pre-processed to control the length within a certain range. Through experimental comparison, we chose to use Batch Normalization to effectively mitigate the impact of abnormal outliers in the trajectory sequence.

We believe that the spatio-temporal information embedded in ship trajectory data is an important feature of its movement process, and there are important temporal attributes in the order of trajectory data. Since Transformer uses an attention mechanism instead of a recurrent neural network, the missing positional information results in the Transformer model belonging to a feed-forward architecture that is insensitive to the order of the inputs. In contrast to the deterministic sine-cosine encoding proposed in the original work , we chose the learnable positional encoding W_{pos}, and allowed it to be trained and learned along with the model. Learnable positional coding has a more prominent performance in many datasets because it can learn the optimal positional representation for a given task. They are projected into a high-dimensional subspace during the learning process, which is different and approximately orthogonal to the subspace where the time series samples are located.

Specifically, we first use a learnable parameter matrix W_{pos} with the shape $[L_{max}, batch\_size, d]$, where L_{max} is the maximum length of all trajectory sequences and d is the embedding dimension of the model. Secondly, we initialize the parameter matrix W_{pos} so that it presents a uniform distribution in the interval [-0.02, 0.02], as shown in the Eq. (2):

$$W_{pos_{i,j,k}} \sim \text{Uniform}(-0.02, 0.02) \tag{2}$$

where i, j and k represent the location index, batch index and embedded dimension index of the trajectory sequence respectively. Then,

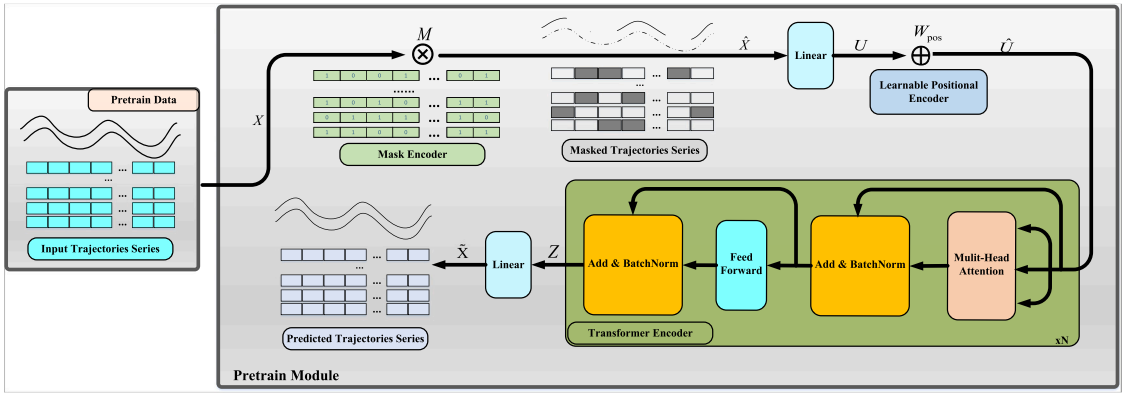


Fig. 4. Architecture of the pretrain module.

for the trajectory sequence X_i of each input model, assuming that the length of trajectory sequence X_1 is L, and the mapping of X_1 through the projection layer is U_1, the encoding of the first L positions of W_{pos} is intercepted to obtain $W_{pos}\text{slice} = \boldsymbol{W_{pos}}[:L,:,:]$. Then U_1 and its corresponding position code $W_{pos}\text{slice}$ are added to get $\widehat{U_1} = U_1 + W_{pos}\text{slice}$. Then, to prevent overfitting of the model, we applied Dropout to the trajectory sequence.

After the above steps are summarized and encoded by learnable position coding, the output of U is shown in the Eq. (3):

$$\hat{\mathrm{U}} = \text{Dropout}\left(\mathrm{U} + W_{pos}[:L,:,:]\right) \tag{3}$$

3.2.2.2. The process of pre-training. Normal trajectory data rarely exhibit sudden changes. If the masked sequence in the trajectory data is very short, it can be approximated by copying the values before and after or by taking the region's average. In order to be able to better learn the interdependencies between the modeled variables, we would like to be able to control the length of the masking sequences instead of simply using some distribution to independently set the noise masks at random. In this paper, we define the proportion of masking as r, which means that in each time step, the value of the average $r * m$ variables will be set to 0. we make the length of each masking segment follow a geometric distribution with a mean of l_α, and the average length of unmasked segments to be $l_\beta = \frac{(1-r)}{r} * l_\alpha$, again conforming to a geometric distribution.

Specifically, suppose that for a given trajectory sequence $X_1 \in R^{L*m}$, we first independently use m all-1 Boolean vectors of length L, obtaining $M = \{M_0, M_1, M_2 \ldots M_{m-1}\}$. Then calculate the relevant parameters:

$$p_m = \frac{1}{l_\alpha}, \quad p_u = \frac{r}{1-r} * p_m, \quad p = [p_m, p_u] \tag{4}$$

in the Eq. (4), p_m is the end probability of the shielded segment, p_u is the end probability of the corresponding unshielded segment, and p is the state transition probability. Then for each mask segment $M_i, i = (0, 1, 2 \ldots m-1)$ with probability r is initialized. After initialization, 0 indicates masking and 1 indicates unmasking, as shown in the following Eq. (5):

$$M_i \sim Bernoulli(r) \tag{5}$$

then, each element of M_i is set separately according to the initialization result. If the state is 0 after initialization, it is transferred to state 1 with probability p_m; if the state is 1 after initialization, it is transferred to state 0 with probability p_u. Repeat the above process m times, and finally get the Boolean mask matrix M with size (L, m).

Considering that each trajectory sample has a different length, the SSRLM model uses padding mask operations in the Transformer Encoder block. For trajectories with a length less than 300, padding is performed with 0 values, and a padding mask is generated. The purpose of the padding mask is to set the attention scores of the padded positions to large negative values, thereby suppressing the attention on the padded positions. This ensures that the model does not focus its attention on the padding marks, thereby improving computational efficiency. Additionally, it helps prevent the model from being affected by sequences of different lengths, enhancing the robustness of the model.

The complete steps of pretraining are as follows: First, perform a masking operation on the trajectory sequence X to obtain the masked sequence $\hat{X} = X \odot M = \left\{\widehat{x_1}, \widehat{x_2}, \widehat{x_3} \ldots \widehat{x_T}\right\} \in R^{T*m}$. Then, linearly project $\hat{X}$ into a vector space of dimension d for the model:

$$\mathrm{u}_t = W_u \widehat{x_t} + \mathrm{b}_u \tag{6}$$

In Eq. (6), $W_u \in R^{d*m}$ and $b_u \in R^d$ are the parameters of the linear projection layer, learned along with the model. $u_t \in R^d, t = [0, 1, \ldots, T]$, is the output of the linear projection layer. Adding positional encoding to u_t yields $\widehat{u_t} = Dropout(u_t + W_{pos})$, which serves as the input vector to the Transformer encoder block. After multiplication by the corresponding matrices, it is used as the query, key, and value for the self-attention layer. The final output vector of the Transformer encoder is represented as z_t. z_t is then fed into a linear projection layer with parameters $W_o \in R^{m*d}$ and $b_o \in R^m$ to output an estimate of the original trajectory sequence. This process is described by Eq. (7):

$$\widetilde{x_t} = W_o z_t + \mathrm{b}_o \tag{7}$$

The output $\widetilde{x_t}$ is the predicted trajectory sequence of the original trajectory sequence x_t. The training loss function utilizes the Mean Squared Error (MSE) between the original trajectory data x_t and the predicted reconstructed trajectory sequence $\widetilde{x_t}$. The formula for the mean squared error loss function for each trajectory sequence is given by Eq. (8):

$$\text{loss}_{\text{ pretrain }} = \frac{1}{T}\sum_{t=1}^{T}\left(\widetilde{x_t} - x_t\right)^2 \tag{8}$$

Consistent with other pre-training models, the process only calculates the error between the predicted values and the actual values for the masked portion of each trajectory sample. No loss function calculation is performed for the unmasked portion.

3.2.3. "One Ship with Multiple MMSI Codes" Recognition Module

The pretrain module utilizes unlabeled trajectory data, extracting dense vector representations of the motion features of ship trajectory sequences. The final linear layer of the Pretrain Module is used to map the vector representation z_t of the Transformer encoder output back to the original feature dimensions. When pretrain module is trained to convergence, it will end the pre-training and freeze all parameters and weights. Then the "One Ship with Multiple MMSI Codes" Recognition Module inherits the previously frozen parameters and weights and use the labeled trajectory data for fine tuning. At this point, the linear

output layer used by the pre-training module will be replaced with a new linear layer, which will be retrained in the fine-tuning phase. The new linear layer is used to map the vector representation z_t of the Transformer encoder output to class numbers. In the fine-tuning phase, due to the settings of activation functions, etc., the model will focus on learning the feature differences between the trajectories of different labels.

First, concatenate the output vectors $z_t \in R^d$ from the Transformer encoder into a new vector $\hat{z} = (z_1, z_2, \ldots, z_T) \in R^{d*T}$. Then, use this concatenated vector as the input to a new linear output layer:

$$\hat{y} = W_k \hat{z} + b_k \tag{9}$$

In Eq. (9), $W_k \in R^{2*d*T}$ and $b_k \in R^2$ are the weights and bias of the new linear output layer, respectively. $\hat{y}$ is the predicted label of whether the trajectory sequence X belongs to the "One Ship with Multiple MMSI Codes" behavior. $\hat{y}$ is obtained from the probability distribution over the class of trajectories via the Softmax function. The loss function for each batch is formulated as shown in Eq. (10):

$$\text{loss}_{\text{train}}(y, \hat{y}) = -\frac{1}{batch\_size} \sum_{c=1}^{batch\_size} y \log \hat{y} \tag{10}$$

the loss function calculates the average cross-entropy between the predicted labels $\hat{y}$ and the true labels y within each batch. This process is calculated using only real number vectors, which can optimize the model efficiency without increasing the amount of computation.

4. Experimental results and analysis

4.1. Dataset and data preprocessing

The raw trajectory data used in this study are offshore ship trajectories centered in Hainan Province, China. The trajectory information is mainly AIS data, which is fused with radar information through networking fusion, and then processed by adaptive threshold algorithm, and each trajectory will be assigned a different ID.

In the process of dataset production, we take a multi-level approach to the label division of ship data. Firstly, we use a rule-based method to screen out a large number of ship trajectory groups with the behavior characteristics of "One Ship with Multiple MMSI Codes". The rule method we adopt is to compare multiple AIS trajectory data within a certain range, calculate their Pearson correlation coefficients at similar moments, and judge whether they are abnormal behaviors. This is also a more widely used method at present.

However, this method has a certain false positive rate, which may misjudge some normal trajectories with similar navigation routes as abnormal trajectories. Therefore, we carried out manual secondary confirmation of these trajectories and carried out label calibration. After that, we further invited the ship management staff to verify these data labels to ensure that the label division of the data has high reliability.

Then, we divide the abnormal behavior recognition of "One Ship with Multiple MMSI Codes" into two scenarios.

The first scenario is to distinguish the normal trajectory group and the "One Ship with Multiple MMSI Codes" abnormal trajectory group, aiming to train a model that directly distinguishes the trajectory of "One Ship with Multiple MMSI Codes". We named the dataset for this scenario HN_MulMI_1, which contains 1454 sets of "One Ship with Multiple MMSI Codes" trajectory samples and 2800 sets of normal trajectory samples.

The second scenario is that the rule is judged as a "One Ship with Multiple MMSI Codes" trajectory group, but the misjudged trajectory group that is manually determined as a normal trajectory is distinguished from the real "One Ship with Multiple MMSI Codes" trajectory group, which aims to train the "One Ship with Multiple MMSI Codes" discrimination model for secondary research and judgment. We named the dataset for this scenario HN_MulMI_2, which contains 251 sets of "One Ship with Multiple MMSI Codes" trajectory samples and 693 sets of misjudged trajectory samples.

Then some samples were randomly selected in all trajectory samples, without labeling and matching. The feature splicing can be directly performed. Through this method we get a total of 15559 groups of unlabeled pre-training samples.

HN_MulMI_1 and HN_MulMI_2 are used for fine-tuning, and they have the same dataset format. Finally, the pre-training dataset, the HN_MulMI_1, and the HN_MulMI_2 together form our "One Ship with Multiple MMSI Codes" anomalous behavior recognition dataset HN_MulMI. It is worth noting that there is no data overlap in these trajectory groups.

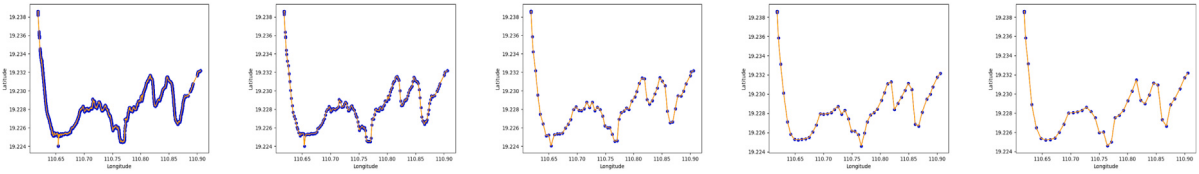
The preprocessing of these AIS data includes grouping the sampled points based on trajectory ID, sorting them by timestamp, and treating them as spatiotemporal sequences; Each trajectory undergoes resampling; Delete the stationary trajectory where the speed features of the whole trajectory are all less than 1; Delete trajectories whose length is less than 10 after resampling; Eliminate other features collected by the sensor, and retain the longitude, latitude, speed, course, heading and time features. Compared with the original data, the preprocessed trajectory data can eliminate redundant information and useless information, which can make the model learn the trajectory feature information better.

For the initial trajectory data, it is redundant in terms of time, with multiple trajectory points with duplicate features often appearing within adjacent time stamps of a few seconds. Therefore downsampling method is needed to eliminate redundant data. Generally, the resampling strategy for time series is to use point-by-point resampling or time resampling to control the density of sampling points. Properly sampled trajectory data not only retains the key information of the trajectory, but also reduces the amount of data processing and improves the robustness of the model. However, considering that the frequency of ship AIS trajectory data acquisition is dynamically adjusted based on the ship's speed, when a ship is stationary or sailing at a low speed, the update frequency of the AIS data may be lower, and therefore the information acquisition is slower. Conversely, when a ship is traveling at high speed or performing maneuvers such as sharp turns, the update frequency of the AIS data may increase to ensure accurate information about the ship's position and status. Therefore, using point-by-point sampling methods may miss the information of the ship in low-speed states and introduce redundancy in maneuvering states, so we adopt the strategy of time resampling.

Fig. 5(a) provides an example of the original trajectory of a ship, with a sailing time from 00:51 to 06:54 on the same day, and a total of 5058 trajectory points collected by the sensors. Fig. 5(b) shows the trajectory after sampling the original trajectory at 30-second intervals, with a total of 557 trajectory points. Fig. 5(c) displays the trajectory after sampling the original trajectory at 1-minute intervals, with a total of 292 trajectory points. Fig. 5(d) illustrates the trajectory after sampling the original trajectory at 2-minute intervals, with a total of 152 trajectory points. Fig. 5(e) shows the trajectory after sampling the original trajectory at 3-minute intervals, with a total of 103 trajectory points.

It can be observed that these trajectories basically retain the relevant features of the original trajectories after resampling. With different intervals of temporal resampling strategies, some of the features of the original trajectories are destroyed in Figs. 5(d) and 5(e). Therefore, the strategy of using a time sampling interval of 2-minute or more is abandoned. Compared to a 30-second sampling interval, trajectories with a 1-minute sampling interval retain the features and shapes of the original trajectories and reduce the time needed for model training. Consequently, we choose 1-minute sampling interval as the resampling criterion.

The format of dataset HN_MulMI is illustrated in Table 1, which is divided into two parts, where the upper table is the format of the unlabeled dataset used for pre-training and the lower table is the format



(a) Original trajectory (b) 0.5-min interval (c) 1-min interval (d) 2-min interval (e) 3-min interval

Fig. 5. The original trajectory and resampled trajectory of the ship.

Table 1
Format of the HN_MulMI dataset. Upper: Pre-training dataset. Lower: Fine-tuning dataset.

Latitude_1	Longitude_1	Speed_1	Course_1	Heading_1	ReceiveTime_1	Latitude_2	...	ReceiveTime_2	id
18.412479	110.262048	4.8	221.1	221	2023/7/14 3:34	18.412673	...	2023/7/14 3:34	0
18.41197	110.261523	2.5	271.1	220.5	2023/7/14 3:35	18.411685	...	2023/7/14 3:35	0
...	...	...	...	...	...	...	...	...	...
18.372111	110.245241	2.7	7.8	7	2023/7/14 5:36	18.373071	...	2023/7/14 5:37	0
18.4751	110.5084	2.9	226.7	226	2023/8/13 19:02	18.47513	...	2023/8/13 19:02	1
...	...	...	...	...	...	...	...	...	...
18.411055	110.261742	3	170	170	2023/7/14 3:36	18.410767	...	2023/7/14 3:37	547
...	...	...	...	...	...	...	...	...	...

Latitude_1	Longitude_1	Speed_1	Course_1	Heading_1	ReceiveTime_1	Latitude_2	...	ReceiveTime_2	id	label
20.239901	109.782452	8.7	122.8	122	2023/8/16 23:19	20.2231	...	2023/8/16 23:19	0	0
...	...	...	...	...	...	...	...	...	...	...
20.27371	110.225786	4.9	117.1	117	2023/8/22 2:55	20.274281	...	2023/8/22 2:55	400	1
20.27314	110.226874	4.5	124.3	124	2023/8/22 2:56	20.273443	...	2023/8/22 2:56	400	1
...	...	...	...	...	...	...	...	...	...	...

of the labeled dataset used for fine-tuning. The "id" column represents the trajectory group to which the content of this row belongs. The "label" column is used to distinguish whether the trajectory group belongs to the "One Ship with Multiple MMSI Codes" behavior. The columns "Latitude_1" to "ReceiveTime_1" and "Latitude_2" to "ReceiveTime_2" represent the features of the two trajectories in the same trajectory group.

4.2. Experimental setup

The experimental setup of this study utilized an Inter(R) Core(TM) I7-7800X CPU @ 3.5 GHz hexa-core CPU and an NVIDIA TAITAN XP graphics processor. The implementation was carried out on a Windows 10 x64-bit operating system using the PyTorch DL framework. The experimental model included a choice of 3 layers of Transformer encoder blocks, with an internal dimension of 128 for the Transformer embedding layer, a dense feedforward part dimension of 256 for the Transformer layer, and 8 heads for multi-head attention. In the pre-training phase, the learning rate was set to 0.001, utilizing the RAdam optimizer, and a batch size of 64. For the "One Ship with Multiple MMSI Codes" recognition task, the base learning rate was set to 0.001, the optimizer selected was RAdam, and the batch size was set to 64. Additionally, a Warmup and Cosine Annealing strategy was employed to schedule the learning rate during the "One Ship with Multiple MMSI Codes" recognition task. The warmup mechanism gradually increases the learning rate from a small value to the preset maximum value at the beginning of training. This helps mitigate early overfitting to mini-batches in the initial stages and contributes to maintaining stability in the deeper layers of the model, the formula is represented as Eq. (11). Cosine annealing is a common learning rate adjustment strategy that avoids the problem that the learning rate drops too fast and leads to unstable training. It also guarantees that the model can maintain a relatively small learning rate in the later stages of training, the formula is explained as Eq. (12). Additionally, a dropout strategy is employed to prevent overfitting, with a dropout rate of 0.10 utilized in both the pre-training and "One Ship with Multiple MMSI Codes" recognition task stages.

$$lr_t = lr_{\text{base}} * \frac{t}{k * c} \tag{11}$$

$$lr_t = lr_{\min} + \frac{1}{2}\left(lr_{\max} - lr_{\min}\right)\left(1 + \cos\left(\frac{T_{\text{cur}}}{T_{\max}}\pi\right)\right) \tag{12}$$

In Eq. (11), lr_t represents the learning rate at the tth iteration, lr_{base} is the base learning rate, and the fraction is used to control the warmup strategy. t denotes the current training step, k is the number of warm-up epochs set, and c is the number of steps required to iterate through the training set once. Typically, $warmup\_step = k * c$. When t is less than $warmup\_step$, the fraction is a value less than 1, causing lr_t to be less than lr_{base}. As training progresses, t increases, lr_t continuously rises until the learning rate lr_t equals lr_{base}, marking the completion of the warm-up phase. In Eq. (12), lr_t is the learning rate at the tth iteration, lr_{max} is the maximum learning rate, lr_{min} is the minimum learning rate, T_{cur} is the current epoch, and T_{max} is the total number of steps.

In order to prevent the SSRLM model from overfitting during the training process, we divide the dataset in the experiment, and divide the pre-training data into training set: validation set: test set in accordance with the ratio of 6:2:2. After disrupting the order of labeled trajectory samples, they are divided into training set: validation set: test set in accordance with the same ratio. It was ensured that there is no data overlap between the training set, validation set and test set.

The evaluation metrics chosen for assessing the performance of model include Precision (P), Recall (R), and F1-Score (F1), whose calculation methods are represented by Eq. (13), Eq. (14), and Eq. (15), respectively. Higher values for these three metrics indicate stronger model performance.

$$Precision = \frac{TP}{TP + FP} \tag{13}$$

$$Recall = \frac{TP}{TP + FN} \tag{14}$$

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{15}$$

TP and FP represent the truly detected anomaly and the error detected anomaly respectively, and FN refers to the normal sample that is misclassified. Precision indicates the proportion of samples with positive predictions that are actually positive. Recall indicates the proportion

of the actual number of positive samples in which the prediction is positive to the proportion of positive samples in the full sample. F1-scroe is the weighted reconciled mean of Precision and Recall for the combined measure of Precision and Recall.

In addition to the above evaluation metrics, we additionally selected Parameters and Avg Time/sample as the other hand evaluation metrics to show the complexity of the model. Where Parameters represents the number of parameters on which the model is trained and Avg Time/sample represents the average time in seconds spent on testing each individual sample.

In order to mitigate the randomness of experimental results, we trained each model five times with different random seeds. Subsequently, these models were tested independently, and the mean and standard deviation of the results from the five tests were calculated. The presentation format of the results is given as Mean ±Standard Deviation(MEAN ±SD).

4.3. Comparison with state-of-the-art methods

To evaluate the performance of the SSRLM model in identifying "One Ship with Multiple MMSI Codes" behavior in maritime trajectories, we compared it with the following DL models:

- MLP (Multilayer Perceptron) (Wang et al., 2017): MLP is a classic neural network based on the biological neuron model. It is one of the most accurate deep neural networks reported in multivariate time series classification (MTSC) (Ismail Fawaz et al., 2019).
- ResNet (Residual Network) (He et al., 2016): ResNet is another highly accurate deep neural network reported in MTSC (Ismail Fawaz et al., 2019). Its architecture consists of multiple residual blocks, and its influence has inspired many subsequent neural network architectures.
- Dlinear (Zeng et al., 2023): Dlinear decomposes time series into trend and remainder series. It applies each of these decomposed series to separate linear networks, models them, and sums the final outputs to predict the result of the time series.
- LightTS (Zhang et al., 2022a): LightTS converts 1D time series into a 2D structure and performs both continuous and interval sampling on 2D time series. It then utilizes MLP to extract features from the structured network.
- PatchTST (Nie et al., 2023): PatchTST is a multivariate time series model based on the Transformer architecture. The key part is to segment the time series data into subseries-level patches, which serve as the input token of Transformer. And each input token contains information from only a single channel.
- TimesNet (Wu et al., 2023): TimesNet is a CNN-based model that transforms one-dimensional time series into a two-dimensional space for analysis. It employs Inception modules and has demonstrated advanced performance in various time series analysis tasks.
- TF-C (Zhang et al., 2022b): TF-C is a pre-trained model for self-supervised contrast learning on time series data, which utilizes the Transformer encoder to generate embeddings in the time and frequency domains and is optimized by contrast loss with the aim of learning a time-frequency consistent representation between the data through pre-training.

Since our comparisons are for the task of "One Ship with Multiple MMSI Codes" trajectory recognition, the inputs to these comparative models are the trajectory sequences X, i.e., the labeled datasets HN_MulMI_1 and HN_MulMI_2, which we have produced for the fine-tuning of the models. Meanwhile, the outputs of these models are the predicted labels $\hat{y}$, which will be compared with the labels y corresponding to the trajectory sequence X, resulting in the various evaluation metrics. If there are models that need to be pre-trained, we will first use our unlabeled pre-training dataset to pre-train them until they converge, and then proceed to the task of recognizing the trajectories of "One Ship with Multiple MMSI Codes".

The specific performance of each model on the dataset HN_MulMI is presented in Table 2. The bolded text in the table indicates the best results. It can be observed that, compared to other models, the proposed SSRLM model demonstrates superior performance in the "One Ship with Multiple MMSI Codes" recognition task in both scenarios, showcasing excellent overall capabilities.

Specifically, because MLP processing trajectory sequence is relatively simple, it is difficult to understand the long-term dependence of trajectory data, so the effect is not good, and MLP model is difficult to make full use of timing information.

By using residual block, ResNet enables a skip connection in the network, which helps to learn complex features of the trajectory sequence, and the design of residual connection helps to extract fine-grained features of the trajectories, which has a better ability to capture the change rule of the trajectory sequence data, so it achieves the third-best and second-best recognition rates in the two scenarios respectively. However, it also has almost the largest number of parameters, resulting in a large amount of computing resources.

Unexpectedly, the Dlinear model yields the poorest results on the HN_MulMI_1 dataset. This is attributed to the vast differences in feature variables among different normal trajectories due to factors such as varying ship traffic, different motion states, and diverse types of ships. Each normal trajectory exhibits its unique motion patterns, while the core architecture of Dlinear is designed to learn the seasonality and trend of variables. It attempts to uniform weight parameters are used to reflect the seasonality and trend of the motion patterns of all normal trajectories, leading to regrettable results and rendering it unable to accurately identify ship trajectory behavior. And in the HN_MulMI_2 dataset, both the misjudgment trajectory group and the "One Ship with Multiple MMSI Codes" trajectory group have their own unified motion characteristics, so the recognition rate of the Dlinear model on this dataset has been greatly increased.

Both LightTS and PatchTST achieve average results on the "One Ship with Multiple MMSI Codes" recognition dataset. LightTS learns potential dependencies between features through a regression-based approach, while PatchTST combines regression and reconstruction methods, utilizing a Transformer structure to learn dependencies between the time and feature dimensions. However, the network structure of the two suggests that they both focus on the individual behavioral characteristics and before-and-after time-series information of each trajectory variable, and to some extent ignore the characteristic differences among all trajectory variables.

The TimesNet model takes into account the multi-periodicity of the time series, reshapes the series in two dimensions, and utilizes the Inception module to simulate intra-periodic and inter-periodic variations of the trajectory series. Although achieving suboptimal results in the scenario of HN_MulMI_1 dataset, it is worth noting that the TimesNet model takes much longer to complete than other algorithms because the number of model parameters is much larger, almost 32 times that of the SSRLM model, and the time required to test a single sample is almost 73 times that of the SSRLM model.

The TF-C model is also used as a pre-training model, and unlike the mask reconstruction approach of the SSRLM model, TF-C focuses on learning a consistent representation in both the time and frequency domains while pre-training. However, its results on Table 2 show that this method does not have a particularly outstanding performance in the field of "One Ship with Multiple MMSI Codes" identification.

The SSRLM model first extracts the trajectory operation laws of ship trajectory sequences through pre-training, learns the interrelationships between the features within the trajectory, reduces the negative impact of the error data collected by the sensors on the recognition of the "One Ship with Multiple MMSI Codes" behavior, and captures the feature dependence and time dependence of the trajectory sequences. After that, the model was fine-tuned to focus on capturing the uniqueness of

Table 2
Performance of each model on the HN_MulMI dataset (MEAN ±SD (%)).

Dataset	Methods	P	R	F1	Parameters	Avg Time/sample
HN_MulMI_1	MLP (Wang et al., 2017)	78.2401 ± 3.9221	77.9523 ± 3.8281	78.0957 ± 3.8728	12 165	0.000186
	ResNet (He et al., 2016)	97.3509 ± 0.2901	97.336 ± 0.3015	97.3434 ± 0.2957	35 376 398	0.000654
	Dlinear (Zeng et al., 2023)	60.6717 ± 2.5540	62.505 ± 2.2787	61.5704 ± 2.3688	192 604	0.000137
	LightTS (Zhang et al., 2022a)	92.4461 ± 3.5184	92.5116 ± 3.4511	92.4788 ± 3.4847	108 124	0.000197
	PatchTST (Nie et al., 2023)	90.4139 ± 1.4027	90.4573 ± 1.4027	90.4356 ± 1.4277	245 828	0.00055
	TimesNet (Wu et al., 2023)	98.1935 ± 0.6954	98.1909 ± 0.6916	98.1922 ± 0.6935	18 826 244	0.035586
	TF-C (Zhang et al., 2022b)	96.9197 ± 1.5015	96.8356 ± 1.5596	96.8776 ± 1.5304	1 643 500	0.000236
	SSRLM(Ours)	**99.2708 ± 0.5883**	**99.2644 ± 0.5931**	**99.2676 ± 0.5907**	590 852	0.000488
HN_MulMI_2	MLP (Wang et al., 2017)	82.1766 ± 1.208	82.109 ± 1.4152	82.1414 ± 1.2578	12 195	0.000196
	ResNet (He et al., 2016)	92.2286 ± 0.5312	92.2275 ± 0.5404	92.228 ± 0.5336	35 376 398	0.00261
	Dlinear (Zeng et al., 2023)	85.5435 ± 4.9904	85.4976 ± 5.089	85.5202 ± 5.0358	190 202	0.000191
	LightTS (Zhang et al., 2022a)	88.148 ± 2.8688	87.9937 ± 2.9009	88.0702 ± 2.8751	113 484	0.000193
	PatchTST (Nie et al., 2023)	87.9736 ± 2.7291	87.6777 ± 2.2975	87.8239 ± 2.4892	551 298	0.000761
	TimesNet (Wu et al., 2023)	90.5083 ± 3.0721	90.237 ± 3.3453	90.3718 ± 3.2032	37 574 914	0.05232
	TF-C (Zhang et al., 2022b)	88.9779 ± 1.7887	89.0995 ± 1.6418	89.0386 ± 1.7153	2 580 100	0.000889
	SSRLM(Ours)	**93.617 ± 1.3165**	**93.4597 ± 1.4761**	**93.5381 ± 1.3917**	514 818	0.000979

the "One Ship with Multiple MMSI Codes" trajectory compared with the normal trajectory, and learning the feature differences between the trajectory variables, which solves the problem of other abnormal conditions that may exist in the trajectory sequence. As can be seen from Table 2, SSRLM not only performs the best recognition rate in two scenarios, but also has more significant advantages in terms of computational cost and efficiency compared to models with similar recognition rates. The above factors show that the SSRLM model has a high performance in recognizing the behavior of "One Ship with Multiple MMSI Codes", and outperforms other models in terms of evaluation indexes.

4.4. The influence of different components and hyperparameters on model performance

To further assess the effectiveness of individual components in the SSRLM model and explore the performance of model selection hyperparameters, several sets of performance analysis experiments were designed in this study. These experiments aimed to observe the impact of changes in model components and corresponding hyperparameters on the detection of "One Ship with Multiple MMSI Codes" behavior. Throughout the experimental process, only components and corresponding hyperparameters were altered between models, while all other parts remained consistent.

- w/o Pretrain: Exclude the pretraining process; directly input the raw trajectory sequence into the d-dimensional vector space for the "One Ship with Multiple MMSI Codes" detection task.
- w/o BatchNorm: Instead of Batch Normalization, use the Layer Normalization that is more commonly used in NLP tasks.
- w/o LearnablePos: Substitute learnable position encoding with absolute position encoding generated by sine and cosine functions.

The results are shown in Fig. 6. The performance of the model decreases significantly after canceling the pre-training task because the pre-training process allows the model to learn more motion characteristics of ship trajectories, and the addition of the pre-training task weakens the negative impact of the error data on the downstream "One Ship with Multiple MMSI Codes" detection task; Batch Normalization has significant performance improvement compared to Layer Normalization because each trajectory sequence data is independent and diverse, Batch Normalization can adapt the model to different data distributions between batches, which helps to deal with the internal covariate drift problem of trajectory sequences in the time dimension. At the same time, Batch Normalization can effectively mitigate the impact of deviation values or abnormal outliers in the actual sampling process of the sensor; Learnable position coding is better than absolute position

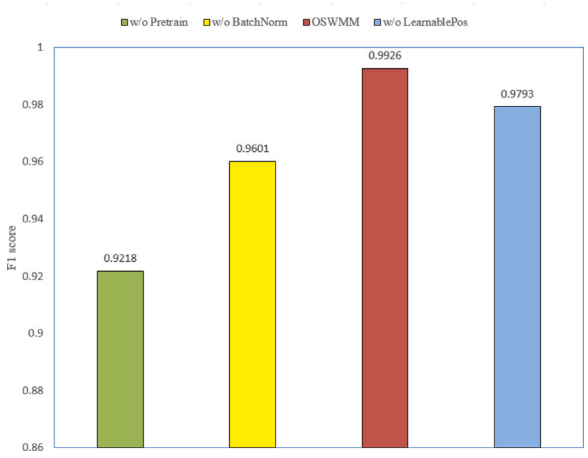


Fig. 6. The influence of different components on SSRLM model on the HN_MulMI_1 dataset.

coding because learnable position coding is very helpful for dealing with trajectory sequences of different lengths, it can adaptively adjust the position coding according to the characteristics of the trajectory sequences, which helps the model focus on identifying more critical information in the task of "One Ship with Multiple MMSI Codes".

After that, we set different mask ratios and different average mask lengths to explore the performance differences of the SSRLM model on the pre-training test set under different mask ratios and different average mask lengths. From the results in Figs. 7 and 8, these can be observed that excessively large mask ratios or average mask lengths can lead the model to lack sufficient context information when predicting masked trajectory information during the pretraining process. The results in the model struggling to effectively learn the motion characteristics of trajectories. Conversely, overly small mask ratios or average mask lengths may cause the model to reduce the amount of information learned from large-scale trajectory data, overlooking the motion characteristics and context information of trajectories. Based on the final experimental results, the model ultimately selected hyperparameters with $l_\alpha = 3$ and $r = 0.15$ as the average mask length and mask ratio for the pretraining module.

4.5. Generalization performance of the model on public datasets

In order to demonstrate the generalization performance of the SSRLM model, we additionally employed three publicly available real-world time series anomaly behavior detection datasets for experimentation. The specific details of the datasets are outlined below.

Table 3
Performance of each model on the public abnormal behavior detection dataset (MEAN ±SD (%)).

Dataset			SMAP		
Metrics	P	R	F1	Parameters	Avg Time/sample
MLP (Wang et al., 2017)	83.1258 ± 3.5169	80.6433 ± 8.6848	81.6354 ± 2.7768	12 240	0.000479
ResNet (He et al., 2016)	82.9975 ± 1.1884	82.2486 ± 5.063	82.5983 ± 3.1442	35 376 398	0.000637
Dlinear (Zeng et al., 2023)	90.6281 ± 0.591	**90.2921 ± 0.5878**	90.2903 ± 0.2984	130 362	0.000431
LightTS (Zhang et al., 2022a)	86.1775 ± 3.7846	85.0872 ± 6.8891	85.6142 ± 5.3578	110 938	0.000452
PatchTST (Nie et al., 2023)	81.6951 ± 1.1297	82.9519 ± 9.798	82.1394 ± 4.2604	418 946	0.000587
TimesNet (Wu et al., 2023)	83.2008 ± 0.1398	81.5956 ± 1.8155	82.386 ± 0.9941	37 509 250	0.003885
TF-C(Zhang et al., 2022b)	89.7036 ± 0.2086	89.0935 ± 3.6936	89.3741 ± 1.9099	531 030	0.000671
SSRLM(Ours)	**90.6695 ± 0.5267**	89.9667 ± 0.0071	**90.3162 ± 0.2618**	409 986	0.000489
Dataset			MSL		
Metrics	P	R	F1	Parameters	Avg Time/sample
MLP (Wang et al., 2017)	91.0762 ± 0.0013	94.7643 ± 0.0291	92.8837 ± 0.0146	12 390	0.000565
ResNet (He et al., 2016)	91.0229 ± 0.0767	92.77 ± 2.8494	91.8779 ± 1.437	35 376 398	0.000891
Dlinear (Zeng et al., 2023)	91.0641 ± 0.0086	94.4902 ± 0.1938	92.7455 ± 0.0978	151 624	0.000653
LightTS (Zhang et al., 2022a)	91.4959 ± 0.1594	92.0573 ± 0.5428	91.7755 ± 0.3499	151 468	0.000712
PatchTST (Nie et al., 2023)	91.241 ± 0.2087	95.1035 ± 0.0727	93.1321 ± 0.0739	441 986	0.000995
TimesNet (Wu et al., 2023)	91.3079 ± 0.0571	92.3845 ± 0.3712	91.8428 ± 0.1925	37 520 770	0.004055
TF-C(Zhang et al., 2022b)	**94.4227 ± 1.424**	90.1008 ± 5	92.1476 ± 2.2449	1 092 750	0.000969
SSRLM(Ours)	92.1408 ± 1.4758	**95.1858 ± 0.0824**	**93.207 ± 0.1167**	413 826	0.000711
Dataset			SMD		
Metrics	P	R	F1	Parameters	Avg Time/sample
MLP (Wang et al., 2017)	92.7929 ± 1.9936	93.291 ± 2.2875	93.0412 ± 2.1398	12 305	0.000747
ResNet (He et al., 2016)	93.6024 ± 0.3554	71.6913 ± 7.3664	81.0969 ± 4.8626	35 376 398	0.000737
Dlinear (Zeng et al., 2023)	94.5792 ± 0.216	90.3301 ± 4.8244	92.3757 ± 2.6282	141 412	0.000513
LightTS (Zhang et al., 2022a)	93.7403 ± 0.0433	63.8499 ± 0.1868	75.9603 ± 0.1179	127 550	0.000563
PatchTST (Nie et al., 2023)	94.6299 ± 0.6713	**95.8995 ± 1.2214**	**95.26 ± 0.9416**	428 930	0.000727
TimesNet (Wu et al., 2023)	94.2618 ± 0.0834	91.5022 ± 1.553	92.8566 ± 0.8061	37 514 242	0.003973
TF-C(Zhang et al., 2022b)	93.3817 ± 4.522	86.8999 ± 7.5125	85.5271 ± 8.8415	770 906	0.000812
SSRLM(Ours)	**94.6902 ± 1.1651**	95.6036 ± 1.5734	95.1445 ± 1.3673	411 650	0.000661

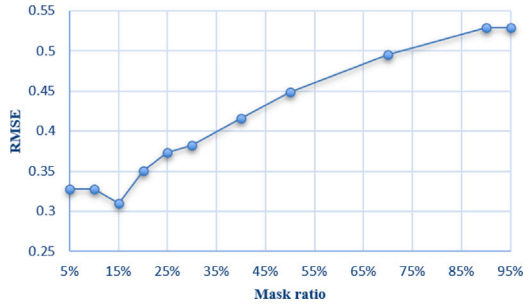


Fig. 7. RMSE on the pretraining test set under different mask ratios.

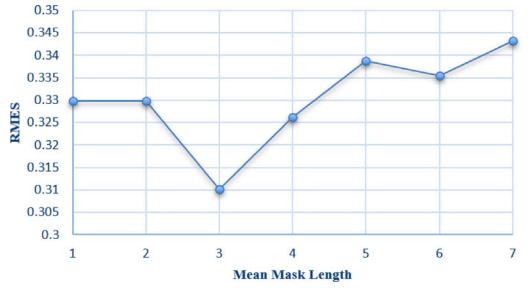


Fig. 8. RMSE on the pretraining test set under different average mask lengths.

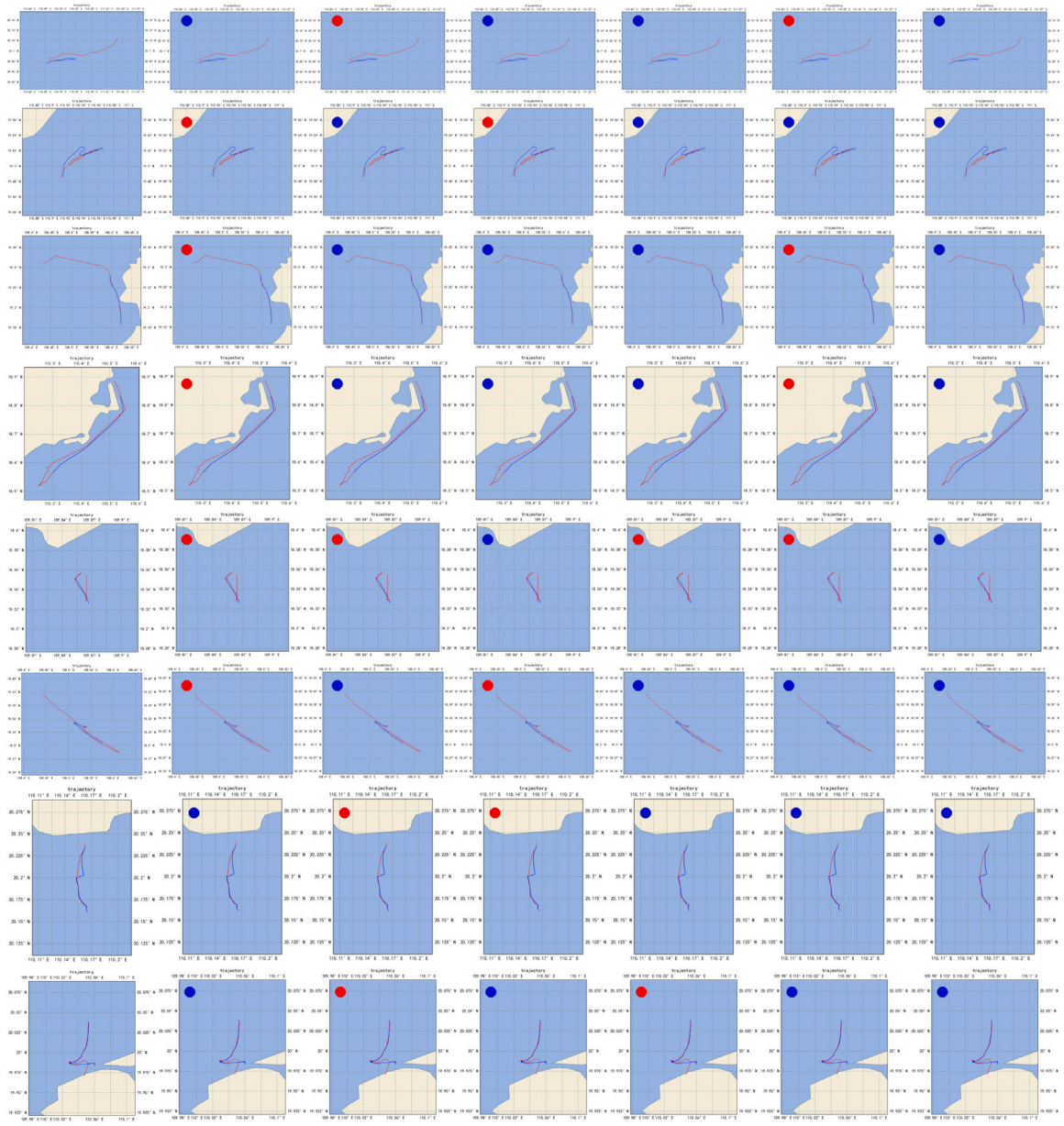
- SMAP(Soil Moisture Active Passive Satellite) (Hundman et al., 2018): This dataset is a spacecraft dataset from NASA, which contains 25 dimensions and is telemetry anomaly data from unexpected anomaly reports of spacecraft monitoring systems.
- MSL(Mars Science Laboratory Rover) (Hundman et al., 2018): The MSL dataset is akin to the SMAP dataset. It corresponds to the sensor and actuator data of the Mars rover itself, totaling 55 dimensions.
- SMD(Server Machine Dataset) (Su et al., 2019): SMD is collected from a large internet company and spans a period of 5 weeks. It records resource utilization from 28 machines in a computing cluster and includes 38 dimensions representing various server metrics.

We followed the same protocol to partition all datasets into training, validation, and test sets. Since the original SMAP, MSL and SMD datasets are unsupervised anomaly detection datasets, we redefined their originally unlabeled training sets and labeled test sets as pre-train dataset and train dataset. Both pre-training data and training data were divided into training, validation, and test sets in a ratio of 6:2:2. Additionally, a sliding window of size 100 was applied to all three datasets. The performance of each model on the test sets is presented in Table 3.

As shown in Table 3, SSRLM model achieves good performance on three different public abnormal behavior detection datasets. Specifically, the SSRLM model proposed in this paper achieves the best R and F1 and the second best P on MSL dataset, the best P and F1 and the second best R on SMAP dataset, and the best P and the second best R and F1 on SMD dataset. In addition, SSRLM also has advantages in terms of the number of parameters and time consumed for testing in all three datasets. Taken together, SSRLM still has the lead in terms of substance, proving that SSRLM model has excellent abnormal behavior detection ability and generalization ability.

4.6. Qualitative experiment

In order to better demonstrate the recognition ability of SSRLM network on the trajectory of "One Ship with Multiple MMSI Codes", we plotted four groups of "One Ship with Multiple MMSI Codes" anomalous trajectories and four groups of normal trajectories, where the main trajectory are colored in blue and the secondary trajectory are colored in red. As shown in Fig. 9(a), in these plots, the first two



(a) Original Images (b) Dlinear (c) LightTS (d) PatchTST (e) TimesNet (f) TF-C (g) **Ours**

Fig. 9. Visualization of the models. The blue circles for correct tests and red circles for incorrect tests.

plots represent tampered MMSI code traces, the 3rd-4th plots represent multiple MMSI code traces turned on, and the last four plots represent normal traces. According to the previous knowledge of maritime personnel, the abnormal trajectory of tampering with MMSI code has the characteristics of the main trajectory suddenly disappearing, and then the secondary trajectory suddenly appearing, and the secondary trajectory inherits part of the motion characteristics of the main trajectory; among the anomalous trajectories with multiple MMSI codes turned on, the secondary trajectory often appears suddenly, and its trajectory is highly similar to a certain section of the main trajectory. The first four plots in Fig. 9(a) fit the above characteristics. Although the fifth and sixth figures in the column 9(a) conform to the characteristics of tampering with the MMSI code, the secondary trajectory of the fifth plot existed long before the main trajectory disappeared; Compared with the main trajectory, the motion characteristics of the secondary trajectory in the sixth figure are quite different. The secondary trajectory of the seventh and eighth plots have independent segments before or after the overlapping segments of the main trajectory, respectively, so they are also normal trajectories.

Subsequently, these eight trajectory groups were entered into each network for testing, and the results were redrawn. The blue circle in the upper left corner means the detection results are correct, and the red circle in the upper left corner means the detection results are wrong. As shown in Figs. 9(b), 9(g), the SSRLM network accurately recognizes the “One Ship with Multiple MMSI Codes” trajectory group. Meanwhile, Dlinear, LightTS, TF-C and PatchTST have different degrees of false alarms and omissions, and TimesNet have a false alarm. The above experimental results demonstrate the superiority of SSRLM network in the field of “One Ship with Multiple MMSI Codes” behavior detection.

At the same time, we believe that the practical application effect of the model is one of its most important criteria. In order to verify the practical ability of the model in real scenarios, we deployed and applied the SSRLM model in the real world.

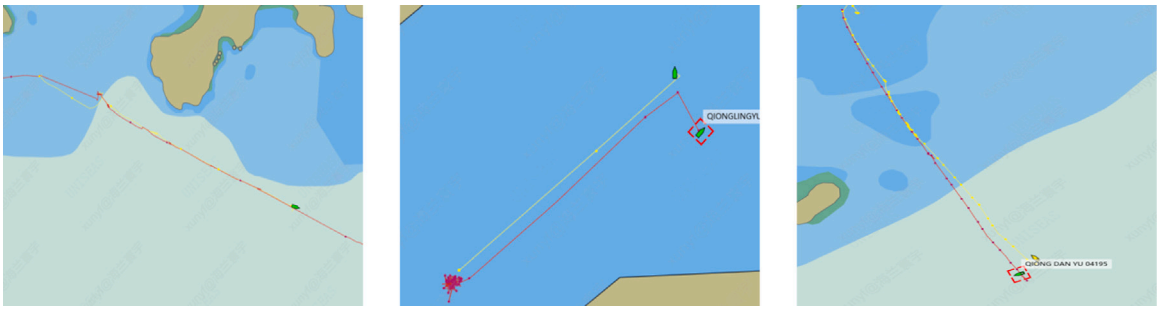


Fig. 10. Trajectories misjudged by the rules are later correctly determined by the SSRLM model.

Taking the data of July 18, 2024 as an example, the rule-based "One Ship with Multiple MMSI Codes" identification method detected 38 groups of abnormal trajectories on that day, and then after the secondary screening of our model, we finally confirmed that the number of "One Ship with Multiple MMSI Codes" abnormal trajectory groups was 9 groups. During this period, there were only 3 false alarms in our model, compared with the 29 false alarms in the rule-based method, our model greatly reduces the false alarm rate and saves human resources consumption. Fig. 10 shows the track sets that were misclassified by the rules as "One Ship with Multiple MMSI Codes" track sets, and then determined to be normal track sets by our model.

5. Conclusions and outlook

The detection of abnormal behavior of ships is of great practical significance in real marine scenarios. The study of "One Ship with Multiple MMSI Codes" behavioral identification is a key link in ensuring shipping safety, effective regulation and prevention of illegal activities, and is essential for maintaining maritime order and protecting the interests of society. It provides new challenges and opportunities for the maritime sector to carry out efficient detection work to maintain maritime safety.

In this study, a dataset HN_MulMI is produced for the problem of "One Ship with Multiple MMSI Codes" behavior recognition using real ship AIS data near Hainan Province, China, and a self-supervised model SSRLM based on the Transformer encoder architecture is proposed. The model learns the motion patterns of ship trajectories through an unsupervised pre-training scheme, which makes full use of easily accessible unlabeled ship trajectories. Afterwards, only a small amount of labeled trajectory data is needed to complete the recognition task of "One Ship with Multiple MMSI Codes", which greatly saves the high cost of labeling datasets. The excellent results of the SSRLM model on the HN_MulMI dataset and in the real-world demonstrate that the structural framework and training methodology of the SSRLM model can adequately capture the deeper features of the "One Ship with Multiple MMSI Codes" trajectory data. Meanwhile, three real multivariate time series datasets from the industrial field are also used for experiments, which demonstrate the excellent generalization ability and anomalous behavior detection capability of the SSRLM model.

At the same time, this research is not only about the specific technical realization, but also aims to optimize the existing detection steps. Through the introduction of intelligent judgment methods of deep learning, the accuracy of the overall work is improved, and the dependence on human experience is reduced, and the early warning of combating illegal acts of ships is improved. Meanwhile, we believe that our study can also provide new ideas for the field of maritime management. Through intelligent judgment, explore how to reduce manual participation, improve the level of automation and intelligence, and provide a reference for building a more intelligent and efficient maritime management system.

Future work will focus on the following aspects: First, collecting more kinds of abnormal trajectory data in real scenarios for multiple types of abnormal ship behavior detection. Second, to correlate the ship trajectory information with the ship's own static characteristic information, such as the ship's own type, height, and draft. Based on the above outlook, we will continue to carry out the work of identifying the trajectories of different kinds of abnormal behaviors of different types of ships.

CRediT authorship contribution statement

Yifeng Sun: Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization,. **Yaochi Zhao:** Conceptualization, Writing – review & editing, Methodology, Validation, Project administration, Data curation, . **Zhuhua Hu:** Supervision, Project administration, Formal analysis, Conceptualization,. **Wei Wu:** Resources, Data curation,. **Jingwen Xia:** Resources, Data curation,. **Yizhen Wang:** Writing – review & editing,.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported by the Key Research and Development Project of Hainan Province, China (Grant no. ZDYF2022GXJS348 and Grant no. ZDYF2024GXJS021), the National Natural Science Foundation of China (Grant No. 62361024 and Grant no. 62161010), and the Hainan Province Natural Science Foundation, China (623RC446). The authors would like to thank the referees for their constructive suggestions.

References

Chen, J., Hu, M., Li, B., Elhoseiny, M., 2022. Efficient self-supervised vision pretraining with local masked reconstruction. arXiv preprint arXiv:2206.00790.

Chen, C.H., Khoo, L.P., Chong, Y.T., Yin, X.F., 2014. Knowledge discovery using genetic algorithm for maritime situational awareness. Expert Syst. Appl. 41 (6), 2742–2753.

Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Eriksen, T., Høye, G., Narheim, B., Meland, B.J., 2006. Maritime traffic monitoring using a space-based AIS receiver. Acta Astronaut. 58 (10), 537–549.

Feichtenhofer, C., Li, Y., He, K., et al., 2022. Masked autoencoders as spatiotemporal learners. Adv. Neural Inf. Process. Syst. 35, 35946–35958.

Gu, Y., Hu, Z., Zhao, Y., Liao, J., Zhang, W., 2024. MFGTN: A multi-modal fast gated transformer for identifying single trawl marine fishing vessel. Ocean Eng. 303, 117711.

Guan, M., Cao, Y., Cheng, X., 2023. Research on the recognition of multiple MMSI codes on a single vessel based on AIS datas. IEEE Access.

Han, P., Wang, W., Shi, Q., Yue, J., 2021. A combined online-learning model with K-means clustering and GRU neural networks for trajectory prediction. Ad Hoc Netw. 117, 102476.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R., 2022. Masked autoencoders are scalable vision learners. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16000–16009.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778.

Hou, Z., Liu, X., Cen, Y., Dong, Y., Yang, H., Wang, C., Tang, J., 2022. Graphmae: Self-supervised masked graph autoencoders. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 594–604.

Huang, Y., Zhang, Q., 2019. Identification of anomaly behavior of ships based on KNN and LOF combination algorithm. In: AIP Conference Proceedings, Vol. 2073, No. 1. AIP Publishing.

Hundman, K., Constantinou, V., Laporte, C., Colwell, I., Soderstrom, T., 2018. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 387–395.

Iphar, C., Napoli, A., Ray, C., 2020. An expert-based method for the risk assessment of anomalous maritime transportation data. Appl. Ocean Res. 104, 102337.

Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., Muller, P.A., 2019. Deep learning for time series classification: a review. Data Min. Knowl. Discov. 33 (4), 917–963.

Kazemi, S., Abghari, S., Lavesson, N., Johnson, H., Ryman, P., 2013. Open data for anomaly detection in maritime surveillance. Expert Syst. Appl. 40 (14), 5719–5729.

Mazzarella, F., Vespe, M., Tarchi, D., Aulicino, G., Vollero, A., 2016. AIS reception characterisation for AIS on/off anomaly detection. In: 2016 19th International Conference on Information Fusion. FUSION, IEEE, pp. 1867–1873.

Murray, B., Perera, L.P., 2020. A dual linear autoencoder approach for vessel trajectory prediction using historical AIS data. Ocean Eng. 209, 107478.

Nie, Y., H. Nguyen, N., Sinthong, P., Kalagnanam, J., 2023. A time series is worth 64 words: Long-term forecasting with transformers. In: International Conference on Learning Representations.

Rhodes, B.J., Bomberger, N.A., Zandipour, M., 2007. Probabilistic associative learning of vessel motion patterns at multiple spatial scales for maritime situation awareness. In: 2007 10th International Conference on Information Fusion. IEEE, pp. 1–8.

Rong, H., Teixeira, A., Soares, C.G., 2020. Data mining approach to shipping route characterization and anomaly detection based on AIS data. Ocean Eng. 198, 106936.

Shao, Z., Zhang, Z., Wang, F., Xu, Y., 2022. Pre-training enhanced spatial-temporal graph neural network for multivariate time series forecasting. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 1567–1577.

Singh, S.K., Heymann, F., 2020. Machine learning-assisted anomaly detection in maritime navigation using AIS data. In: 2020 IEEE/ION Position, Location and Navigation Symposium. PLANS, IEEE, pp. 832–838.

Soares, A., Dividino, R., Abreu, F., Brousseau, M., Isenor, A.W., Webb, S., Matwin, S., 2019. Crisis: Integrating ais and ocean data streams using semantic web standards for event detection. In: 2019 International Conference on Military Communications and Information Systems. ICMCIS, IEEE, pp. 1–7.

Su, Y., Zhao, Y., Niu, C., Liu, R., Sun, W., Pei, D., 2019. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 2828–2837.

Terroso-Saenz, F., Valdes-Vela, M., Skarmeta-Gomez, A.F., 2016. A complex event processing approach to detect abnormal behaviours in the marine environment. Inf. Syst. Front. 18, 765–780.

Tuli, S., Casale, G., Jennings, N.R., 2022. Tranad: Deep transformer networks for anomaly detection in multivariate time series data. arXiv preprint arXiv:2201.07284.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. Adv. Neural Inf. Process. Syst. 30.

Vesecky, J.F., Laws, K.E., Paduan, J.D., 2009. Using HF surface wave radar and the ship automatic identification system (AIS) to monitor coastal vessels. In: 2009 IEEE International Geoscience and Remote Sensing Symposium, Vol. 3. IEEE, pp. III–761.

Wang, Y., Liu, J., Liu, R.W., Liu, Y., Yuan, Z., 2023. Data-driven methods for detection of abnormal ship behavior: Progress and trends. Ocean Eng. 271, 113673.

Wang, Z., Yan, W., Oates, T., 2017. Time series classification from scratch with deep neural networks: A strong baseline. In: 2017 International Joint Conference on Neural Networks. IJCNN, IEEE, pp. 1578–1585.

Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J., Long, M., 2023. TimesNet: Temporal 2D-variation modeling for general time series analysis. In: International Conference on Learning Representations.

Wu, X., Rahman, A., Zaloom, V.A., 2018. Study of travel behavior of vessels in narrow waterways using AIS data – A case study in Sabine-Neches Waterways. Ocean Eng. 147, 399–413.

Xiao, F., Ligteringen, H., van Gulijk, C., Ale, B., 2015. Comparison study on AIS data of ship traffic behavior. Ocean Eng. 95, 84–93.

Xu, J., Wu, H., Wang, J., Long, M., 2021. Anomaly transformer: Time series anomaly detection with association discrepancy. arXiv preprint arXiv:2110.02642.

Yin, C., Zhang, S., Wang, J., Xiong, N.N., 2020. Anomaly detection based on convolutional recurrent autoencoder for IoT time series. IEEE Trans. Syst. Man Cybern. A 52 (1), 112–122.

Zeng, A., Chen, M., Zhang, L., Xu, Q., 2023. Are transformers effective for time series forecasting? In: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 37, No. 9. pp. 11121–11128.

Zerveas, G., Jayaraman, S., Patel, D., Bhamidipaty, A., Eickhoff, C., 2021. A transformer-based framework for multivariate time series representation learning. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. pp. 2114–2124.

Zhang, W., Goerlandt, F., Kujala, P., Wang, Y., 2016. An advanced method for detecting possible near miss ship collisions from AIS data. Ocean Eng. 124, 141–156.

Zhang, T., Zhang, Y., Cao, W., Bian, J., Yi, X., Zheng, S., Li, J., 2022a. Less is more: Fast multivariate time series forecasting with light sampling-oriented mlp structures. arXiv preprint arXiv:2207.01186.

Zhang, T., Zhao, S., Cheng, B., Chen, J., 2020. Detection of ais closing behavior and mmsi spoofing behavior of ships based on spatiotemporal data. Remote Sens. 12 (4), 702.

Zhang, X., Zhao, Z., Tsiligkaridis, T., Zitnik, M., 2022b. Self-supervised contrastive pre-training for time series via time-frequency consistency. In: Proceedings of Neural Information Processing Systems, NeurIPS.

Zhao, L., Shi, G., 2019. Maritime anomaly detection using density-based clustering and recurrent neural network. J. Navig. 72 (4), 894–916.

Zhen, R., Jin, Y., Hu, Q., Shao, Z., Nikitakos, N., 2017. Maritime anomaly detection within coastal waters based on vessel trajectory clustering and Naïve Bayes classifier. J. Navig. 70 (3), 648–670.