Faculty of Mathematics and Informatics

VILNIUS UNIVERSITY
FACULTY OF MATHEMATICS AND INFORMATICS
DATA SCIENCE
MASTER'S STUDY PROGRAMME

EVALUATING MARINE VESSEL COLLISION RISKS USING TRAJECTORY PREDICTION REGIONS AND CLUSTERS

JŪRŲ LAIVŲ SUSIDŪRIMO RIZIKOS ĮVERTINIMAS NAUDOJANT TRAJEKTORIJOS SPĖJIMO REGIONUS IR KLASTERIUS

Master's Thesis

Author: Kasparas Kaukas
VU email address: kasparas.kaukas@mif.stud.vu.lt
Supervisor: Dr. Julius Venskus

Vilnius

2024

Abstract

This paper develops and assesses a comprehensive framework for collision risk using trajectory prediction regions and clusters. Using month's worth of Automatic Identification System (AIS) data from the Danish Maritime Authority, RNN based models were created to predict the further trajectory of marine vessels. Ten noised predictions were utilized to create prediction regions. To evaluate collision risk, we implemented both boundary-based methods (ellipsoidal point intersection, ellipsoidal area intersection, and Z-score bounding box area intersection) and non-boundary-based methods (Gaussian Mixture Model (GMM) area intersection and a novel silhouette value-based score transformed via root functions). Results indicated that boundary-based methods performed poorly, particularly due to the "pass-by problem," where non-simultaneous region intersections fail to capture collisions effectively. In contrast, non-boundary-based methods, particularly GMM, demonstrated superior performance due to their continuous nature. GMM detected collisions with high accuracy, including identifying a real collision scenario up to 20 minutes prior to the event. These findings highlight the limitations of boundary-based metrics and underscore the potential of continuous, probabilistic approaches for improving maritime safety.

Keywords: vessel trajectory prediction, collision risk, ellipsoidal prediction region, Z-score, Gaussian mixture model, Silhouette value

Santrauka

Šiame darbe kuriama ir įvertinama visapusė susidūrimo rizikos sistema, naudojant trajektorijos prognozavimo regionus ir klasterius. Naudojant Danijos jūrų laivybos tarnybos mėnesio automatinės identifikavimo sistemos (AIS) duomenis, buvo sukurti RNN modeliai, skirti numatyti tolesnę jūrų laivų trajektoriją. Kuriant prognozavimo sritis buvo panaudota dešimt triukšmingų prognozių. Norėdami įvertinti susidūrimo riziką, naudojome tiek ribomis pagrįstus metodus (elipsių taškų sankirtos metodas, elipsių plotų sankirta ir Z-įverčio ribinių langelių plotų sankirta), tiek ne ribomis pagrįstus metodus (Gauso skirstinių mišinio modelio (GMM) srities sankirta ir naujas silueto įverčiu pagrįstas metodas, transformuotas naudojant šaknines funkcijas). Rezultatai parodė, kad ribomis pagrįsti metodai veikė prastai, ypač dėl „prasilenkimo problemos“, kai vienu metu vykstančios regionų sankirtos nesugeba veiksmingai užfiksuoti susidūrimų. Priešingai, neribiniai metodai, ypač GMM, parodė puikų našumą dėl tolydaus pobūdžio. GMM susidūrimus aptiko labai tiksliai, įskaitant tikro susidūrimo scenarijaus nustatymą likus 20 minučių iki įvykio. Šios išvados išryškina ribomis pagrįstų metodų apribojimus ir pabrėžia tolydžių, tikimybinių metodų, skirtų gerinti jūrų saugą, potencialą.

Raktiniai žodžiai: laivų trajektorijų spėjimas, susidūrimo rizika, elipsių spėjimų regionai, Z-įvertis, Gauso skirstinių mišinio modelis, Silueto įvertis.

Contents

1 Introduction

Even though past years have brought many adverse economic variables to many countries around Europe, maritime transport has still held the top spot of freight transport in the EU accounting for 67.8% of total cargo transport [12] in the European Union in 2022 totaling in 3.43 billion tonnes of cargo shipped [13]. Such high demand for maritime activity poses risks of rare but costly vessel collisions. Not only do collisions pose a problem of damaged vessels, and lost and damaged cargo but such accidents are one of the main causes of maritime human casualties. From an annual overview, it is apparent that collisions are the main casualty event type from 2014 to 2022 with 21.6% of the occurrences [2]. This highlights the critical importance of addressing maritime safety to reduce the risks associated with high levels of maritime transport activity.

Considering the importance of the problem and favorable conditions (existence of Automatic Identification Systems (AIS)) it is clear that this problem can be tackled using modern approaches such as neural networks which can be used for trajectory prediction region formation which then can be employed in methods that evaluate the intersections between them giving collision risk score.

Thus the goal of this thesis is to evaluate marine vessel collision risks using trajectory prediction regions and clusters.

To achieve this objective, the research is structured around a series of tasks designed to systematically address the evaluation of collision risks and the development of effective methodologies:

1. A literature review is conducted to examine existing research and methodologies in the fields of trajectory prediction, collision risk assessment, and prediction regions, providing a foundation for identifying gaps and opportunities for further exploration. Appropriate scientific methods for trajectory prediction and collision risk evaluation are identified and selected based on their relevance and suitability.

2. A detailed methodology is formulated to offer a theoretical basis for the methods utilized in the experimental segment of this research.

3. An experiment is carried out involving trajectory prediction, the formation of trajectory prediction regions, and the application of collision risk evaluation methods. A comparison of the methods is performed to evaluate their effectiveness and suitability.

4. A real-case analysis is conducted to validate the applicability and effectiveness of the proposed methods in practical scenarios.

2 Literature overview

Having AIS data freely accessible has caught the science community's attention and has been a push to develop methods for trajectory prediction. A variety of different techniques and approaches to solving trajectory prediction problems have been visualized in the article "AIS data-driven ship trajectory prediction modeling and analysis based on machine learning and deep learning methods" [10]. A cluster analysis was done to highlight the main solution areas for trajectory predictions. 67 papers between 2000 and 2022 were filtered out as relevant and they were clustered into 6 ideas - recurrent neural

networks, attention mechanism, deep learning, meta-model based simulation optimization, machine learning, and real-time systems. Further, these papers were put into 3 general groups - papers based on motion characteristics, papers based on machine learning, and papers based on deep learning methods.

Though not popular, trajectory prediction using motion characteristics can be easily interpreted as it uses dynamic and static AIS data directly (e.g. derived speed, rate of turn, course over ground, type of ship, etc.) to predict further movement of the vessel based on similar ship activity in the area [9]. Moreover, as the model created by the authors is quite general, it could also be used in anomaly detection. Even though, accuracy-wise, the model performs quite well in short-range prediction it lacks precision in calculating long-range trajectories. Overall, while such models are appealing in interpretability and low computational expenses, they are primarily suited for very specific environmental scenarios, and for high accuracy, ideal conditions have to be met.

To overcome stated issues, statistical approaches have been used more commonly, many of them falling under the machine learning category. In the paper "Framework for Ship Trajectory Forecasting Based on Linear Stationary Models Using Automatic Identification System" [23] the authors have employed a popular Auto-Regressive Integrated Average Model (ARIMA). Such a model is highly applicable in a time-series prediction context as it uses dependency between an observation and a set of lagged observations, and the residual errors obtained by applying a moving average model to the lagged sequences. The developed model showed high precision of trajectory prediction (RMSE of 0.023 and 0.017) using 48 hours as the prediction window.

Another possible, though not as popular for time-series predictions, approach is using tree-based algorithms to predict trajectories based on other ships' past trajectories. One such approach is using Random Forests (RF) [28]. In this paper, the authors develop an RF model with a port frequency-based decision strategy to predict the frequencies of vessel travels between two different ports. Such research though different is also useful to better understand vessel movement dynamics where ports are used as one of the metrics to evaluate the precision of prediction.

In the past few years, deep learning approaches have been used more often than any other approach [29]. One such approach is using a recurrent neural network (RNN), which has been proven to be highly useful in many applications such as handwriting [14] and speech recognition [18]. RNN has also been used in predicting trajectories for marine vessels [25] as it is highly applicable in predicting time-series data as it uses sequence data as an input. In this paper, authors develop a novel RNN-based framework that is modified by using gated recurrent units (GRU) which deals with the main RNN issue - the vanishing gradient problem. This problem appears while extending sequence length and was first discovered by Josef Hochreiter in 1991. To overcome this problem, the RNN architecture was modified to include mechanisms for better managing long-term dependencies. LSTM networks incorporate special units called memory cells that can maintain information for long periods, along with gating mechanisms that control the flow of information into and out of these cells. These modifications enable LSTMs to effectively overcome the vanishing gradient problem, thereby improving their performance on tasks involving long sequences of data.

Uncoincidentally, such networks are prevalent in vessel trajectory prediction. Looking at an overview of related work [29] on this topic we find that the first use of LSTM for the medium was done in 2018 [16]. In this paper, the authors choose to use LSTM as an alternative to RNN due to its computational

shortcomings. The architecture of the model chosen is the Encoder-Decoder type. Such architecture is comprised of two multilayered LSTM networks. First LSTM block (Encoder) maps each sequence to a state vector which summarises the whole source sequence. Then second LSTM (Decoder) uses the state vector and last sequence to generate target sequence this way predicting future movement of the vessel. To evaluate, authors use log perplexity - a measure of uncertainty and it manages to reach 1.44, achieving better results than other models. Another variation of the LSTM network is Bidirectional LSTM. The main difference of such variation is that the network not only considers past but future data as well. Though mainly such functionality is useful for natural language processing, it has been also used for predicting vessel trajectories with high accuracy [6] by adjusting model's parameters in real time with existing data as an input. Another way of improving predictions of trajectories is by using attention based mechanisms. Such functionality has been applied in the paper "Ship Trajectory Prediction Based on Attention in Bidirectional Recurrent Neural Networks" [27]. Use of such a model is useful as attention based mechanism applies weights to the inputs this way leveraging comprehensive context captured by the bidirectional processing. Also, applying weights to the inputs could help with the interpretability of the model. As captured by the authors, using attention based Bi-LSTM has a low offset in the predictions compared to other models.

Recently there has been a move towards something called hybrid methods which take advantage of two or more different methods. One such approach used LSTM with historical filtering and cubic spline interpolation [5]. In this study, authors incorporate cubic spline interpolation to use it as a multi-step predictor. These predictions are being created on 3 types of points - start, support, and destination. The destination point is generated by historical data while the support point is created by a trained LSTM. This way predictions are kept in high accuracy for both short and long distances.

While trajectory prediction serves as a foundational component of this research, the accurate prediction of collision events remains a critical focus, as it directly addresses the safety and risk assessment objectives of maritime navigation.

Liu et al. [11] proposed an anchorage collision risk model that combines geometric probability indicators—Distance at Closest Point of Approach (DCPA), Time to Closest Point of Approach (TCPA), and Ship Domain Overlapping Index (SDOI)—with a new global indicator called "safe room," reflecting the navigable space within an anchorage. Using AIS data from anchorages off the Shandong Peninsula, the model effectively identified collision risks in dense anchorage zones. Unlike open-sea models, which focus on active navigation, this approach adapts to the stationary and low-mobility nature of anchored vessels, providing valuable insights into localized risk assessment.

Another study [24] employed the Quaternion Ship Domain (QSD) integrated with the Non-Linear Velocity Obstacle (NLVO) algorithm to develop a model for real-time ship collision risk detection. By redefining the conflict position based on ship dimensions, maneuverability, and encounter scenarios, the model improved upon traditional velocity obstacle methods. Simulations were conducted for three common maritime scenarios: head-on, crossing, and overtaking situations. The QSD-NLVO model demonstrated superior accuracy in detecting collision risks compared to conventional NLVO algorithms, successfully identifying risks in cases where the latter failed. Results showed that the proposed model effectively detected collision risks and provided early warnings, with risk durations observed at 76, 90, and 218 seconds for the head-on, crossing, and overtaking scenarios, respectively.

The paper titled "Mahalanobis Distance-Based DBSCAN for Vessel Near Collision Detection" [3] introduces a novel approach to enhance maritime safety using the DBSCAN clustering algorithm tailored with the Mahalanobis distance metric. The research uses Automatic Identification System (AIS) data to identify potential near-collision scenarios among vessels. Traditional DBSCAN using Euclidean distance often fails to capture the nuanced relationships and varied distributions within AIS data, which Mahalanobis distance effectively addresses by considering data correlation and covariance.

Though implicit, anomaly detection has been utilized as an indirect measure to assess potential collision events. In the study "Unsupervised marine vessel trajectory prediction using LSTM network and wild bootstrapping techniques" [26] authors extend their own previous studies on anomaly detection by applying wild bootstrapping techniques to introduce noise into the data, thereby creating prediction regions used for identifying anomalies. While this paper focuses on single-vessel anomaly detection, it does not extend its scope to interactions between multiple vessels and thus does not explicitly address collision scenarios.

In summary, the reviewed literature provides valuable insights, particularly in the domain of trajectory prediction, showing the effectiveness of advanced techniques such as RNN-based models, including LSTMs, GRUs, and other neural network architectures. Furthermore, the collective body of work highlights a critical gap in the evaluation and comparison of boundary based and non-boundary based collision risk assessment methods in the open sea, proving the need for further research to address this important aspect of maritime safety.

3 Methodology

3.1 Data

For this thesis, Danish AIS data has been chosen [4]. In short, AIS is a VHF-based navigation and anti-collision tool making it possible to exchange information between ships. This information is collected in a shore-based AIS system operated by the Danish Maritime Authority.

The data date range is chosen to be 31 days period from September 14th, 2024 to October 14th, 2024. It was selected due to recency and applicability reasons for modeling as for neural network approach, big data is preferred.

To have a concise prediction medium the data has to be cleaned.

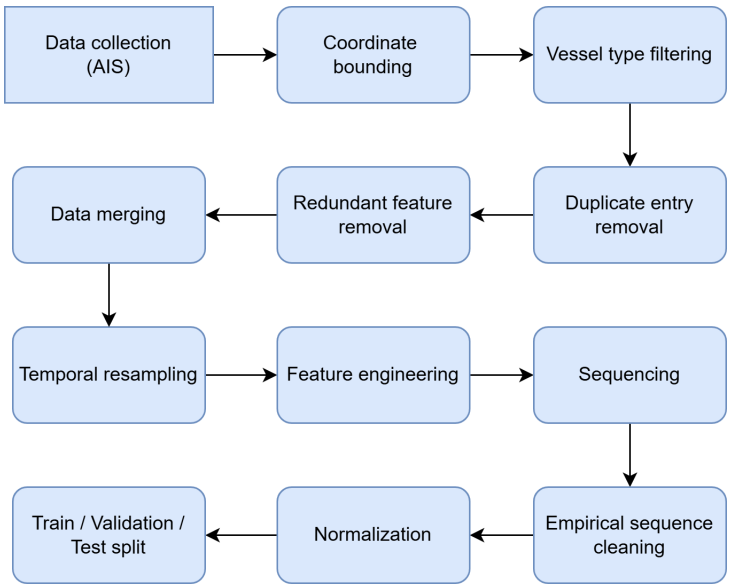


Figure 1: Data preprocessing flow

Data preprocessing starts at a file level (one day). A high-traffic bounding box has been chosen for both simplification of the dataset and to have a clearer view of the region of interest, as well as, simple filtering of vessel type is being made to fit the study goal. Duplicate removal is conducted to remove misreported records. To further optimize the dataset removal of irrelevant and uninformative columns is done, leaving only the most important temporal, spatial, and auxiliary features in the set.

After joining all files into one, similarly to the paper "Application of coordinate systems for vessel trajectory prediction improvement using a recurrent neural networks" [8], temporal resampling is done to mitigate the issue of irregular time points.

Besides data cleaning, a simple feature engineering is done. It includes creating columns for date differences between subsequent vessel data points, differences in latitude and longitude, latitude and longitude speeds, and haversine distance between subsequent records.

Lastly, as the neural network approach is chosen for this work, normalization of each column is necessary, thus a simple MinMaxscaler is used to fit and transform the data, which after trajectory prediction is done, will have to be reversed.

3.2 Sequencing

As trajectory prediction is a spatiotemporal task, the methods to solve it require a special preparation of the data. Firstly, we have to choose the length of a sequence that we are interested in, in our case a similar approach to [8] is selected, 50 minute window is chosen where the first 30 time steps will be the predictor variable and the coordinates of the last 20 minutes will be the target variable. The sequences will be created going through each vessel's data with a step of size 25, meaning the overlap of data in sequences will also be of size 25.

After transforming the data into such format, the final cleaning steps of sequences need to be done. It involves, taking only sequences where all time intervals are of one minute, where the ship doesn't stop for the whole sequence, but also where the vessel doesn't move for more than 771 meters in one

timestep as the maximum speed of cargo ships is 771 meters per minute. Such cleaning steps are necessary for combating cases where vessels' subsequent records do not follow the natural behaviour of the ship's movement. Though most cases are being filtered out before sequencing, the interest region bounding box provides cleaning for the data where vessels exit the bounding box and return in two subsequent records.

Lastly, The dataset was partitioned into training, validation, and test sets with a 70%, 15%, and 15% split, respectively. This allocation is a common practice in machine learning, aiming to provide sufficient data for model training while reserving adequate portions for validation and testing to assess model performance and generalization.

3.3 Trajectory prediction

For this thesis, a neural network approach is used, as a way to predict the future trajectory of a moving vessel. For this task, recurrent neural network variations are chosen as they are designed to train and predict data that is reshaped into a sequential format. Sequential data predictions are highly applicable in the context of time series.

To choose the best model a combination of different parameters is used to train the models. After the choice is made for the best model, the chosen model will be trained 10 times on ten different variations of train and validation data (test data will be kept the same) meaning a random state split of data will be chosen keeping the proportion of the split 75% and 15% accordingly for each train iteration this way creating noise in predictions. Such a decision will be crucial for collision predictions as prediction regions will be formed.

3.3.1 Recurrent neural network (RNN)

Though not used in this thesis experiment, it is necessary to understand the fundamental architecture of a recurrent neural network as we will use modified versions of such a network [15].

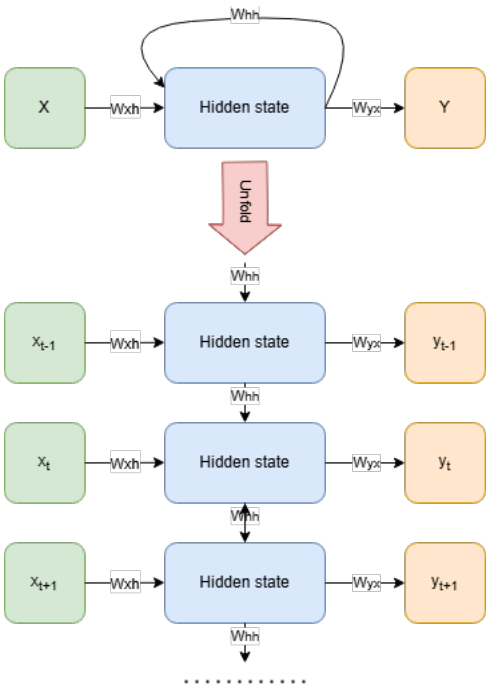


Figure 2: Recurrent neural network architecture

RNNs are most commonly used with sequential data as input, which requires the input to be three-dimensional. The dimensions represent the number of sequences, the number of timesteps in each sequence, and the number of features per timestep. This input structure will be consistently applied across all neural networks used in this thesis.

The main part of an RNN is the recurrent unit which holds memory, that is being updated at each time step based on the current input and previous memory (hidden state). Though useful, RNNs have one main flaw, for which they are not as popular today. It is often called the vanishing gradient problem or exploding gradient problem. It arises when either network depth or sequence length is too long, thus at every training step, when each network weight is being updated with a new value that is proportional to the partial derivative of the loss function, the gradient decreases (vanishes) or increases (explodes) uncontrollably making the learning process slow down (sometimes stop) or overshooting the minima and the learning process never ends accordingly.

To combat this problem, many turn to a variation of an RNN which can deal with such an issue.

3.3.2 Long short-term memory (LSTM)

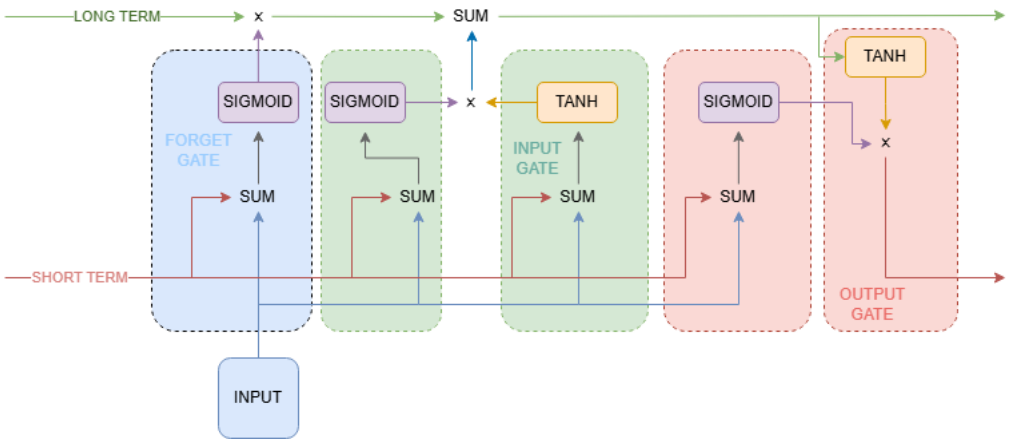


Figure 3: LSTM architecture

Unlike standard RNNs, LSTMs solve the vanishing gradient problem through an internal gating mechanism, enabling them to learn long-term dependencies [15]. Such network has the same input structure as RNNs 3.3.1.

Internally, a single LSTM cell is constructed from:

- **Cell state** (C_t): A memory that flows through the network and retains relevant information over long sequences.
- **Hidden state** (h_t): The output of the LSTM at each time step, representing the information available to the next layer.

Each LSTM cell processes information at a specific time step, updating the cell state and hidden state based on its inputs and internal computations.

The functionality of an LSTM cell is controlled by three primary gates 3 that handle the flow of information:

- **Forget gate** determines what information to discard from the cell state. It uses a sigmoid activation function to calculate a "forget" vector:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

  where:

  - f_t is the forget gate output
  - W_f and b_f are the weights and biases for the forget gate.
  - $[h_{t-1}, x_t]$ combines the previous hidden state (h_{t-1}) and the current input (x_t)

- **Input gate** determines which new information should be added to the cell state. It consists of two steps:

– **Input modulation:** Calculates candidate values to add to the cell state:

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1,x_t} + b_C])$$

– **Gate control:** determines the importance (percentage) of candidate values:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

The cell state is then updated:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

- **Output gate** controls the information passed to the next hidden state:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

The hidden state is then updated.

$$h_t = o_t \cdot \tanh(C_t)$$

The combined operations of these gates allow LSTMs to effectively retain, update, and output information, making them ideal for modeling sequential data with both short and long-term dependencies. By adjusting the flow of information through the cell state and hidden state, LSTMs solve the problems of traditional RNNs, such as the vanishing gradient problem, and enable robust learning of spatiotemporal tasks.

3.3.3 LSTM autoencoder (AE)

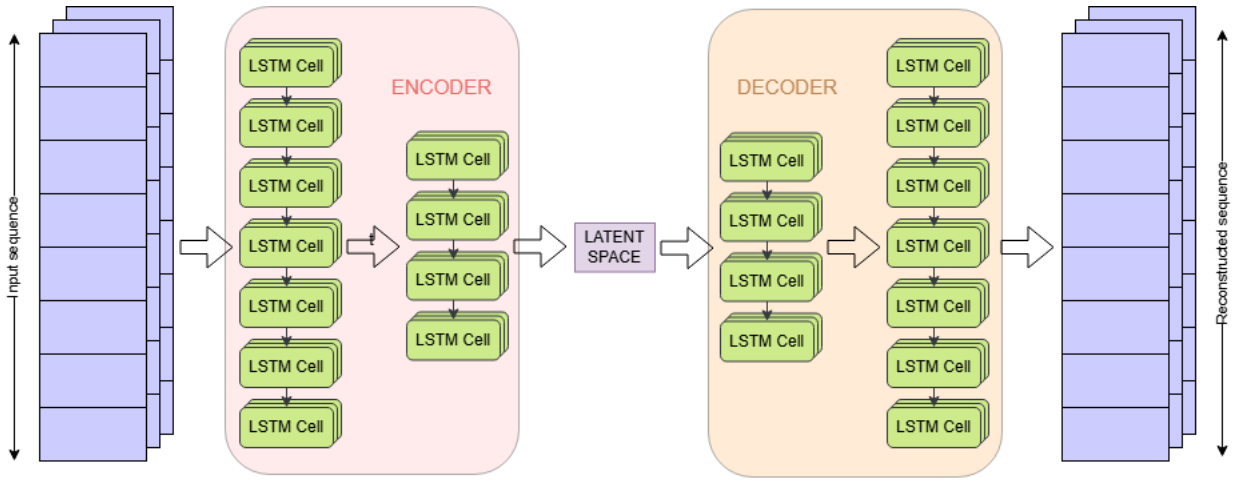


Figure 4: LSTM Autoencoder architecture

The use of LSTMs as building blocks allows the autoencoder to use their ability to model long-term dependencies in sequential data, as detailed in the previous section 3.3.2. LSTM AE architecture consists of 3 main parts [21]:

- **Encoder** is composed of one or more LSTM layers that process the input sequence $(x_1, x_2, \ldots, x_T)$, generating a fixed-length context vector z that summarizes the input. This vector captures temporal dependencies and relevant features of the input sequence.
- **Latent space**: the context vector (z) serves as a compressed representation of the input sequence, providing a bottleneck where it identifies and keeps the most important features of the sequence.
- **Decoder**, also built with LSTM layers, takes the latent vector (z) as input and attempts to reconstruct the original sequence. The decoder mirrors the encoder structure, ensuring alignment between the latent representation and the reconstructed sequence

Unlike a regular LSTM, the LSTM autoencoder focuses on unsupervised learning by reconstructing input sequences. Such reconstruction-based approach allows it to learn compact and important representations of data, making it well-suited for tasks of spatiotemporal predictions.

3.3.4 Bi-directional LSTM

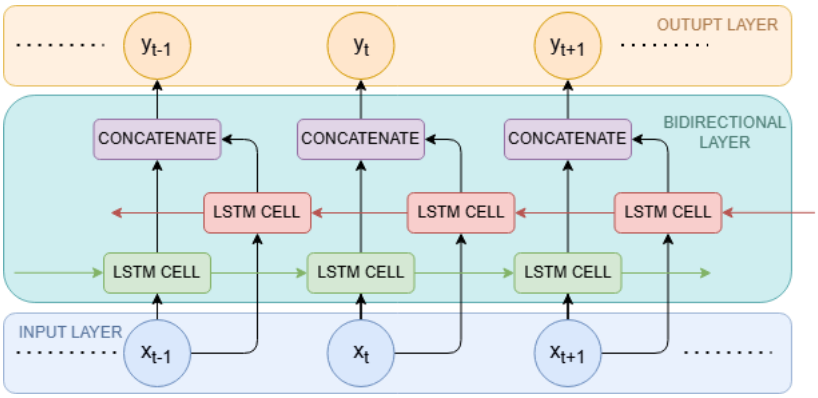


Figure 5: Bidirectional LSTM architecture

A Bidirectional Long Short-Term Memory (Bi-LSTM) network extends the capabilities of a standard LSTM by processing input sequences in both forward and backward directions [15].

This architecture enables the model to capture dependencies from both past and future contexts, making it particularly effective for tasks where full-sequence information is essential The BiLSTM architecture consists of two LSTM layers operating in parallel:

- **Forward Layer:** Processes the input sequence in chronological order, learning dependencies from past to future.
- **Backward Layer:** Processes the input sequence in reverse order, learning dependencies from future to past.

At each time step, the outputs from both layers are concatenated or combined to form the final representation:

$$h_t = [h_t^{\text{forward}}, h_t^{\text{backward}}]$$

where

- h_t^{forward} is the output of the forward LSTM layer.
- h_t^{backward} is the output of the backward LSTM layer.

Such dual perspective ensures that the model has a complete understanding of the entire sequence.

3.3.5 Gated recurrent unit (GRU)

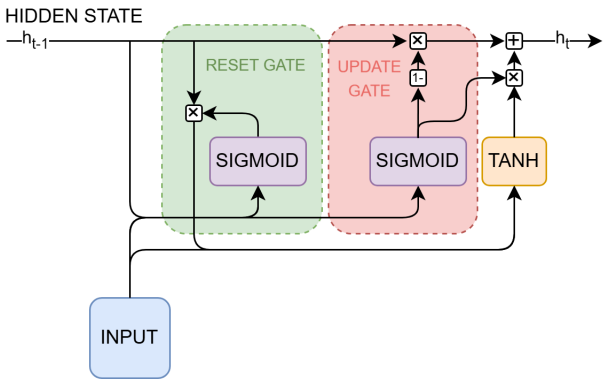


Figure 6: Gated recurrent unit architecture

Gated Recurrent Units (GRUs) are a simplified variant of Long Short-Term Memory (LSTM) networks, designed to model sequential data efficiently while addressing the vanishing gradient problem [15]. GRUs achieve this through a gating mechanism similar that to LSTM's, which reduces computational complexity compared to LSTMs while maintaining strong performance.

A GRU cell consists of two gates:

- **Reset gate** controls the influence of the previous hidden state (h_{t-1}) on the current candidate state:

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

  where

  - r_t is the reset gate output
  - W_r and b_r are the weights and biases for the update gate.
  - $[h_{t-1}, x_t]$ combines the previous hidden state and the current input.

- **Update gate** determines how much of the past information is carried to the next time step

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

Using reset gate's influence, candidate state ($\tilde{h}_t$) is calculated:

$$\tilde{h}_t = \tanh(W_h \cdot [r_t \cdot h_{t-1}, x_t] + b_h)$$

Lastly, the final hidden state (h_t) is then updated using a combination of the candidate state and the previous hidden state, weighted by the update gate:

$$h_t = z_t \cdot h_{t-1} + (1 - z_t) \cdot \tilde{h}_t$$

By using update and reset gates to regulate information flow, GRUs capture both short and long-term dependencies in sequential data offering a more efficient alternative to LSTMs combining the benefits of gated mechanisms with reduced computational complexity.

3.4 Collision prediction

As mentioned in the trajectory prediction section 3.3, 10 predictions are being made for noise to create prediction regions. This means that ten slightly different output sequences are being produced for each input sequence. For each time step, such predicted points can be clusterized into regions which later can be compared to other clusters to evaluate the pseudo-probability of vessel collision. To evaluate said probability, the following methods will be applied and compared.

3.4.1 Haversine distance

Haversine distance is a distance measure between two points on the surface of a sphere, assuming a spherical Earth model. It is widely used in geospatial applications due to its ability to measure great-circle distances, making it particularly suitable for maritime scenarios.

The Haversine distance d between two points $A(lat_1, lon_2)$ and $B(lat_2, lon_2)$ on the Earth's surface is given by formula [17]:

$$d = 2r \arcsin\left(\sqrt{sin^2\left(\frac{\Delta lat}{2} + \cos(lat1) \cdot \cos(lat2) \cdot \sin^2\left(\frac{\Delta lon}{2}\right)\right)}\right)$$

where:

- r is the radius of the Earth (mean radius $r = 6371$km)
- $\Delta lat = lat_2 - lat_1$ is the difference in latitudes in radians
- $\Delta lon = lon_2 - lon_1$ is the difference in longitudes in radians
- $lat_1, lat_2, lon_1, lon_2$ are the latitudes and longitudes of the two points (converted to radians).

In this study, the Haversine distance is used to minimize computational costs by filtering out first cases where predictions are highly improbable. Logically, we will disregard cases where the collision is impossible due to the speed limitations of vessels. This will help identify cases where vessel trajectories are close enough to warrant detailed collision prediction analysis which will be done by more sophisticated methods.

3.4.2 Ellipsoidal prediction region

Before comparing two clusters, for ease of calculation and interpretable visual representation, the ellipsoidal prediction region is chosen. It is built similarly to [1] by choosing a prediction interval value to be 95% meaning there will be a 95% probability that the prediction will be in the ellipse region. As the prediction is bi-variate (latitude and longitude), a covariance matrix needs to be constructed, which is then used to construct a prediction ellipse also known as the error ellipse.

Mathematically any rotated ellipse can be expressed as a parametric equation:

$$\begin{cases} x(t) = r_x \cos(t)\cos(\theta) - r_y \sin(t)\sin(\theta) \\ y(t) = r_x \cos(t) sin(\theta) + r_y \sin(t)\cos(\theta) \end{cases} \quad t \in (0, 2\pi)$$

Here r_x and r_y denotes vertex and co-vertex lengths, θ - angle between x - axis and vertex. More conveniently we can reform the parametric equation into a matrix form:

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} r_x \cos(t) \\ r_y \sin(t) \end{bmatrix}$$

This can be easily transformed to a confidence ellipse using eigenvectors v_1, v_2 and eigenvalues λ_1, λ_2 from the covariance matrix.

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} v_{1x} & v_{2x} \\ v_{1y} & v_{2y} \end{bmatrix} \begin{bmatrix} \sqrt{\lambda_1} \cos(t) \\ \sqrt{\lambda_2} \sin(t) \end{bmatrix}$$

Lastly, to ensure the confidence ellipse encompasses 95% of the probability mass, the cumulative chi-square distribution value for two degrees of freedom is utilized, where $\chi^2_2(5.991) = 0.95$ and the ellipse is moved to the center of predictions:

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} v_{1x} & v_{2x} \\ v_{1y} & v_{2y} \end{bmatrix} \begin{bmatrix} \sqrt{5.991\lambda_1} \cos(t) \\ \sqrt{5.991\lambda_2} \sin(t) \end{bmatrix} + \begin{bmatrix} \overline{x(t)} \\ \overline{y(t)} \end{bmatrix}$$

Ellipsoidal region was chosen due to scattered distribution shape of predictions which happens to be more scattered in one direction (width).

3.4.3 Ellipsoidal point intersection

The ellipsoidal point intersection metric is a custom measure developed by the author to quantify the overlap between two point distributions (clusters) by evaluating their spatial relationship within defined confidence ellipses. This metric provides an intuitive measure of how much the points in one cluster are spatially aligned with the region of another cluster. By assessing mutual overlap in 95% confidence ellipses, the metric captures the interaction between the two distributions.

To calculate whether a point is inside an ellipse we will use a formula:

$$(x - \mu)^T \cdot (s \times cov(points))^{-1} \cdot (x - \mu) \leq 1$$

where:

- x - point of interest
- μ - mean point of distribution (center)
- s - chi-squared value, as we are working with 95% ellipse, we will apply 5.991 as this value as distribution is of 2 degrees of freedom.
- $cov(points)$ - covariance matrix of distribution points. It was also used in creating the ellipse prediction region.

If the the calculated value is lower or equal to 1 then it is considered that the point is inside the ellipse, otherwise - outside.

Further, to calculate the ratio of points that are intersecting other vessel's ellipse formula is used:

$$EPI(X,Y) = \frac{c(X_Y) + c(Y_X)}{c(X) + c(Y)}$$

where:

- X, Y are the cluster representations where each cluster consists of coordinate points
- $c(X_Y)$ and (Y_X) are the counts of how many points from cluster X lies inside cluster's Y confidence ellipse and vice versa accordingly.
- $c(X)$ and $c(Y)$ are the counts of how many are in each cluster

Such metric not only is highly interpretable but also as it is statistically rigorous as it makes use of confidence ellipse this way incorporating statistical spread and orientation of clusters.

3.4.4 Ellipsoidal area intersection

The ellipsoidal area intersection metric measures overlap between two ellipsoidal confidence regions, representing the prediction distributions of vessel trajectories. The overlap is calculated using the Intersection over Union (IoU) of the areas of the two ellipses.

The ellipses are constructed using the covariance matrix detailed in 3.4.2.

To calculate the intersection region area, integral is calculated:

$$A_{\text{intersection}} = \int_{x_{\min}}^{x_{\max}} \int_{y_{\min}}^{y_{\max}} 1 \, dy \, dx$$

In the experiment, the numerical trapezoidal method was chosen to evaluate this integral.

3.4.5 Z-score

The Z-score is a statistical measure that represents the number of standard deviations (σ) a data point is from the mean of a distribution. In this study, the Z-score is utilized to create bounding rectangles around prediction clusters, where predefined Z-scores determine the dimensions of the rectangles.

For each cluster, the Z-score rectangle requires mean of the clusters latitude μ_{lat} and longitude μ_{long} separately as well as the standard deviations accordingly ($\sigma_{lat}, \sigma_{long}$)

To better understand the possibilities of collisions two Z-scores are chosen - 2 and 3. Thus, the according widths and heights of the rectangular regions will be 4σ and 6σ. More generally formulas for width and height are:

$$W = 2Z \cdot \sigma_{lon} \qquad\qquad H = 2Z \cdot \sigma_{lat}$$

To calculate the score of the collision, intersection over union (IoU) is chosen:

$$\text{IoU} = \frac{\text{Area of Intersection}}{\text{Area of Union}}$$

The use of Z-score-defined rectangular regions is a more conservative approach compared to ellipsoidal regions, as it gives broader coverage by taking into account possible outliers within the boundaries of the rectangle. Such approach prioritizes safety, which is critical for collision prediction.

3.4.6 Gaussian mixture model

The Gaussian Mixture Model (GMM) is a probabilistic model used to represent data as a mixture of multiple Gaussian distributions [19]. In this study, the GMM is applied to model the spatial distributions of vessel trajectories for latitude and longitude separately. To assess the likelihood of collision between two vessels, the kernel density estimates (KDEs) of their latitude and longitude distributions are constructed, and their intersections are computed as joint probabilities. Kernel density estimation (KDE) is a non-parametric method to estimate the probability density function of a variable. For latitude and longitude distributions ($f_{\text{lat}}(x)$ and $f_{\text{long}}(x)$) of each vessel:

$$f(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

where:

- n is the number of total points in the distribution
- h is the bandwidth (smoothing parameter) which is set by Scott's rule:

$$n^{-\frac{1}{d+4}}, \qquad n - \text{number of points}, \qquad d - \text{number of dimensions}$$

- K is the kernel function, in this case - Gaussian kernel:

$$K(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{u^2}{2}}$$

To get the likelihood of collision, overlaps of both latitude and longitude are calculated:

$$I_{\text{lat}} = \int_{-\infty}^{\infty} \min\left(f_{\text{lat},1}(x), f_{\text{lat},2}(x)\right) dx, \qquad I_{\text{lon}} = \int_{-\infty}^{\infty} \min\left(f_{\text{lon},1}(y), f_{\text{lon},2}(y)\right) dy$$

The joint probability of interaction between the two vessels is calculated as the product of the latitude and longitude intersection probabilities:

$$P_{\text{interaction}} = I_{\text{lat}} \cdot I_{\text{lon}}$$

3.4.7 Silhouette value

The Silhouette value is a metric used to evaluate the quality of clustering by quantifying how well a data point fits within its assigned cluster compared to other clusters [20]. In this study, clusters represent prediction regions that group vessel trajectories or regions of potential collision risk. It ranges from -1 to 1, where higher values indicate well-separated clusters. In the context of vessel collision prediction, the Silhouette value is employed to assess the separation between two prediction clusters (representing regions of vessel trajectories). Well-separated clusters indicate low collision risk, as the prediction regions of the vessels are distinct, minimizing the likelihood of collision. On the other hand, overlapping or poorly separated clusters suggest a higher risk of collision, as the prediction regions of the vessels are closely interacting.

For a point i in a cluster, the Silhouette value s(i) is calculated as:

$$s(i) = \frac{b(i) - a(i)}{max(a(i), b(i))}$$

where:

- $a(i)$: The average distance of point i to all other points in the same cluster (intra-cluster distance).
- $b(i)$: The minimum average distance of point i to all points in the nearest cluster (inter-cluster distance)

The overall Silhouette score for the clustering is the mean of Silhouette values across all points:

$$S = \frac{1}{N} \sum_{i=1}^{N} s(i)$$

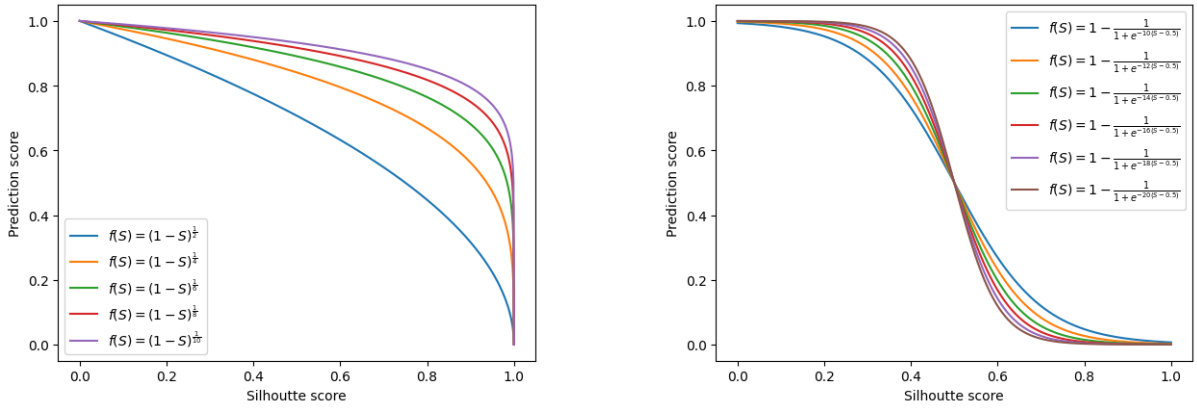
where N is the total number of points in the dataset.

As the Silhouette score gives values between -1 and 1, it becomes difficult to compare this value to other metrics used in this paper. To overcome this problem we will apply a novel custom function to the silhouette score. Firstly, as the silhouette value gets below zero and gets close to -1, it means that the points inside a cluster get closer to other cluster's points than to its own, in our case as the clusters are predefined, it is very unlikely for such scenario to happen thus the custom score function will only be evaluated between silhouette value 0 and 1 and if it goes below 0, the function will be evaluated to 1 - very high probability of collision.

To better understand the best fitting custom function, variations of two functions were chosen:

- $(1 - S)^{\frac{1}{n}}$ - power function with $n >= 1$. In such function as n increases the collision scoring will get more conservative, especially in cases where the silhouette score is very high.

- $1 - \frac{1}{1+e^{-n(S-0.5)}}$ - custom logistic function. In this function, it is important to choose sufficient n so that when S approaches 0 or 1 the score should be approximately 1 and 0 accordingly. Also, this function could be further customized, for example, if needed the mid-point could be moved from 0.5 to another value to achieve a result that would make the model more strict or conservative. Another way could be a split of function at the midpoint, this way value n could be changed independently, achieving different rigidity, however, such variations are only suggestive and require further research and are out of scope for this paper.



(a) Prediction scores using different power transformations of silhouette value

(b) Prediction scores using different logistic transformations of silhouette value

Figure 7: Prediction scores using silhouette value

4 Experiment

In this section, we apply the methodologies discussed earlier to conduct the experimental study. Initially, we preprocess the data to ensure it meets the requirements for model training and collision prediction methods.

4.1 Data preparation

The data used in this thesis is retrieved from the Danish Maritime Authority [4]. For our experiment, a 31 day period has been chosen from September 14th, 2024 to October 14th, 2024. Such a period consists of 31 files for each day. Each file varies from 16,091,453 to 39,474,177 rows summing all up to a total of 655,843,081 rows. In addition, the data is explained by 26 columns - Timestamp, Type of mobile, MMSI, Latitude, Longitude, Navigational status, ROT, SOG, COG, Heading, IMO, Callsign, Name, Ship type, Cargo type, Width, Length, Type of position fixing device, Draught, Destination, ETA, Data source type, A, B, C, D. Later, as we will omit some of the features, we will explain the meaning of each of them.

To overcome the size of the data, before the concatenation of files, a simple data cleaning was conducted for each file. Firstly, only vessel records inside a high-traffic area in the Baltic Sea were chosen. The area of interest chosen spans from 54 to 56 latitude and from 12 to 15 longitude:

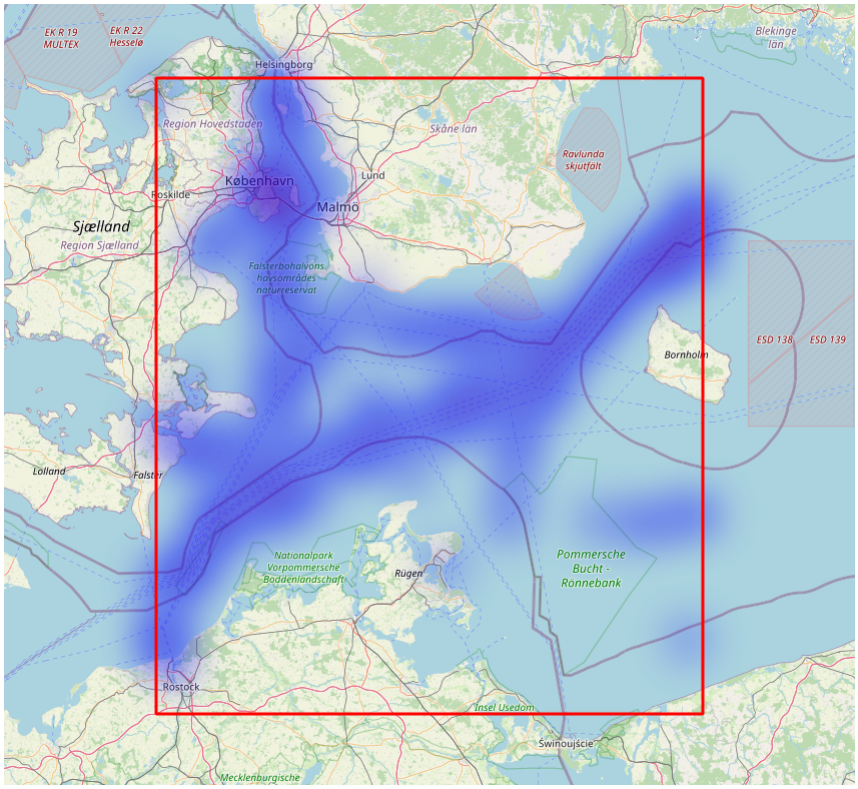


Figure 8: Coordinate bounding box with vessels movement heatmap

To further clean the data only cargo type ships were chosen as this is our zone of interest for this research.

As the data is gathered with different transmitter types, irregularities are often documented, thus duplicates are found in the dataset, to overcome this we are omitting all duplicates of MMSI and Timestamp combination so Timestamp and MMSI together are always unique.

Lastly, for data cleaning, we will only keep columns for **MMSI**, **Timestamp**, **Latitude**, **Longitude**, **SOG**, **Heading**. Such a choice was made due to importance of data availability of these features. MMSI, Timestamp, Latitude, and Longitude are the main columns for identification, temporal, and spatial data definition. SOG and Heading were chosen as auxiliary columns that define speed and direction at any point in time for any vessel. Most of the omitted features were either serving the same purpose as the chosen ones (IMO, Callsign and Name are all identificational data types as they serve the same purpose as MMSI), are metadata of vessels that is irrelevant for trajectory and collision predictions (Type of mobile - Describes what type of target this message is received from, Ship type - as all chosen ship types are cargo this variable becomes irrelevant, Cargo type, Width, and Length, Type of position fixing device, destination, ETA, Data source type), data that is highly sparse (ROT - rate of turn, COG - course over ground, Draught, A, B, C, D - are the lengths from GPS to a different part of the ship itself, also counted as irrelevant) or misleading (only Navigational status falls for this one as the values don't correspond to the real movement of the ship e.g. the value is Moored but the ship is moving, to overcome this SOG, Heading columns will be used as well as feature engineering).

However, before creating new columns, the data has to be resampled. It is due to records being updated in a very varied way where the difference between two records of one vessel could be from 1

to 10 seconds or even bigger when the ship is moored/not moving. To prevent this issue a one minute interval has been chosen where the vessels will be resampled. This means that the Timestamp will take on the format of DD/MM/YYYY HH24:MI omitting seconds. This will be done by choosing the closest record to a particular minute. After resampling, we might end up with cases where either two consecutive records are at the same point (vessel didn't move or the next record is too far away time-wise) or two consecutive points are logically too far spatially (this can happen when the sample for the next time point is not created properly) both of these issues will be addressed in sequencing part of the data.

After resampling, new features are created:

- Date_diff - date difference between two consecutive records.
- Lat_lag - lagged latitude feature (previous latitude for vessel)
- Long_lag - lagged longitude feature (previous longitude for vessel)
- Lat_speed - Latitudal speed ($\frac{\text{Lat}-\text{Lat\_lag}}{\text{Date\_diff}}$)
- Long_speed - Longitudinal speed ($\frac{\text{Long}-\text{Long\_lag}}{\text{Date\_diff}}$)
- haversine_distance - the distance between two consecutive coordinates.

These features will be helpful not only for modeling trajectories but also before modeling to remove illogical records.

As we are modeling with variations of recurrent neural networks, the data has to be adjusted accordingly - made into sequences. Following [8], sequence of length 50 was chosen, in other words, the sequence shows 50 minute window of one ship's movement. It is split into 30 and 20 minute splits where one is used as predictor data and other as target accordingly. After the sequences are created, each one is cleaned so that only sequences without spaces timewise are kept, also we are taking out sequences where there are instances of vessels that moved more than 771 meters in one minute (subsequent records) as 25 knots (771 meters per minute) is considered to be the maximum speed of any cargo ship, there might be exceptions however due to abundance of data we can disregard these cases. Lastly, if there is a total halt in movement of a ship we are omitting such sequence ($Lat_{t-1} = Lat_t$ and $Long_{t-1} = Long_t$)

After sequencing, data normalization is applied to the data. Mathematically:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Lastly, we shuffle the sequences in a controlled random state and the data is split into train, test, and validation sets (70%, 15%, 15%). It is important to note that we also keep duplicate shuffled sequences with identifying columns such as MMSI and Timestamp as in the working data set they are omitted.

4.1.1 Mock collision generation

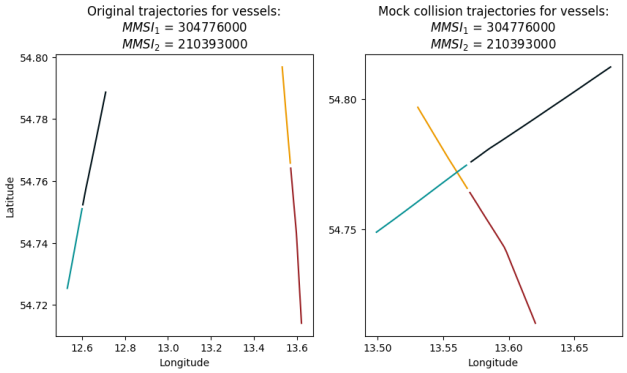
To better understand collision prediction likelihood, simulated collision events are generated and referred to in this paper as 'mock collisions'. Single mock collision is produced by taking two vessels

that have any of the sequential target variable timepoint match and moving their sequence spatially to have a matching coordinate point at a single timepoint.

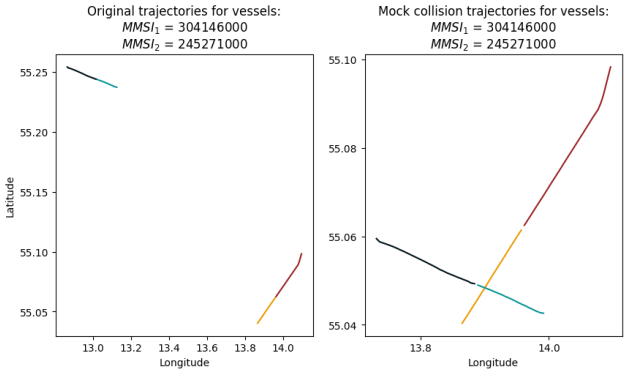
To have a better visual representation as well as a logically sufficient way to evaluate prediction score 10 collisions were generated.

The decision for the collisions was made empirically with a goal to make them varied, including cases where collision would occur perpendicularly, head-on, with a curve, etc.

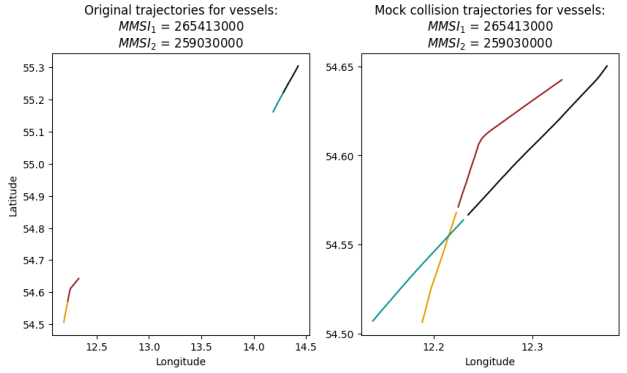
In the visualizations below first vessel's predictor sequence is colored a darker blue color whereas the target sequence is a lighter blue. Similarly for the second ship where color tones are of red color, darker, and lighter correspond for predictor and target sequences accordingly:



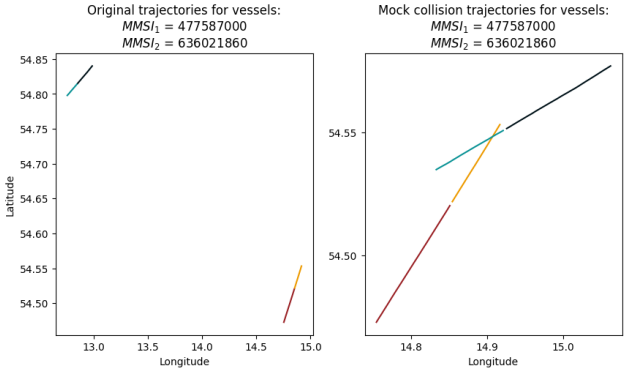
(a) Mocked collision 1



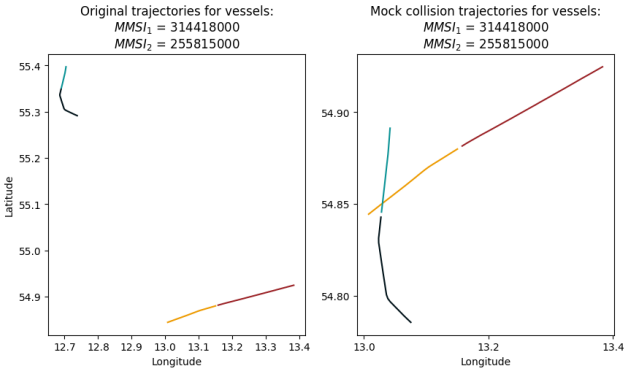
(b) Mocked collision 2



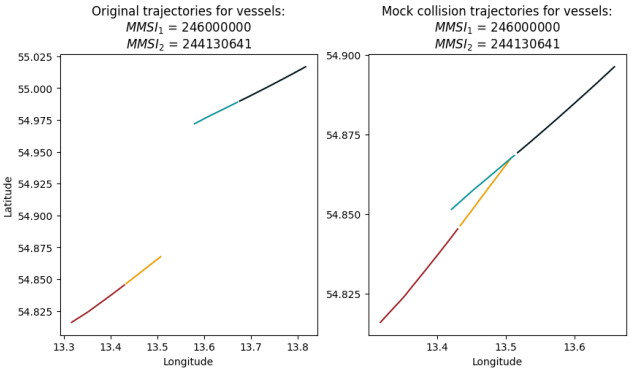
(c) Mocked collision 3



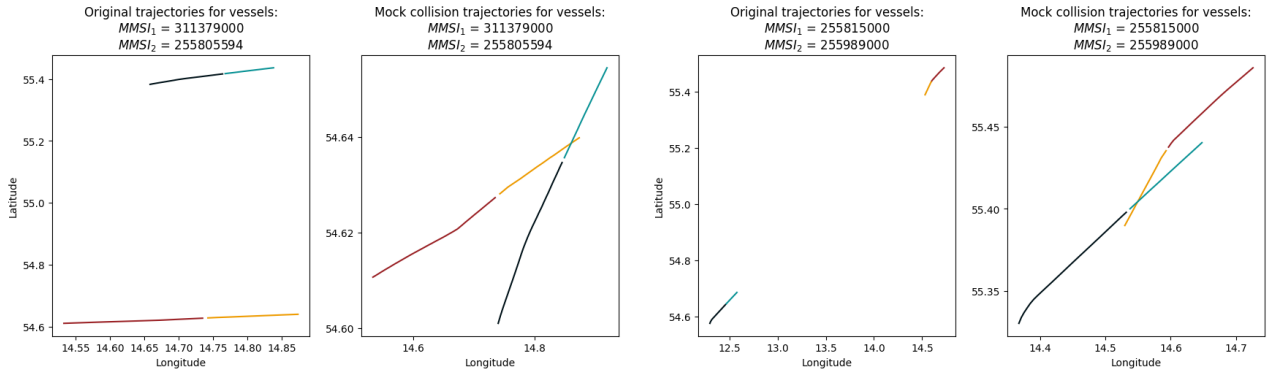
(d) Mocked collision 4



(e) Mocked collision 5

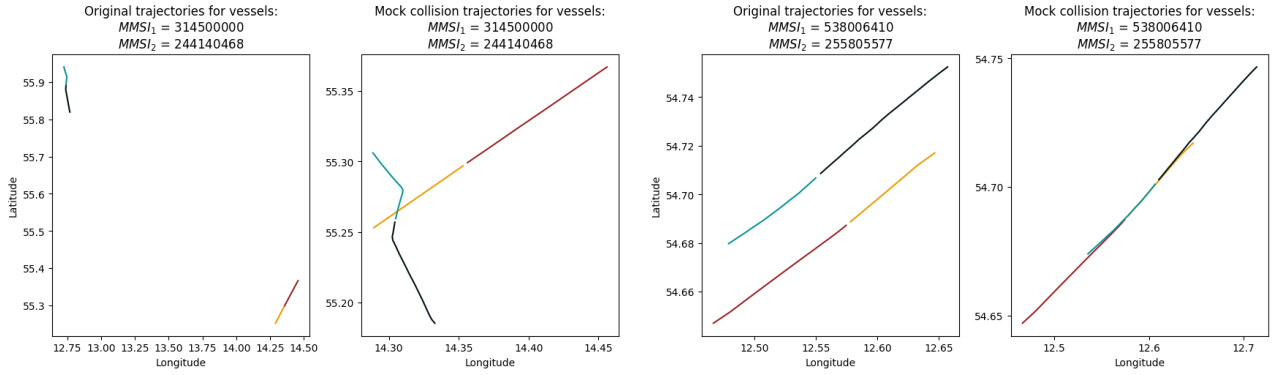


(f) Mocked collision 6



(g) Mocked collision 7

(h) Mocked collision 8



(i) Mocked collision 9

(j) Mocked collision 10

Figure 9: Visualizations of mocked collisions

4.1.2 Filtering

Having a total of 10505 sequences means that in total we will have $10505^2 = 110355025$ two vessel interactions. However, this number can be lowered as going through all of the interactions is computationally expensive.

Firstly we can apply combinations formula $C(n, 2) = \frac{n(n-1)}{2}$ as the order of interaction does not matter as well as self-interactions between sequences are excluded. Such simple filtering makes the number of sequence interactions to be:

$$C(10505, 2) = \frac{10505(10505 - 1)}{2} = 55172260$$

Such number is almost half of the first interaction count but is still extensive.

To further reduce this number, we will take out interactions between vessels where the same vessel is present as this is a logical fallacy. After removing self-vessel interactions 55105983 interactions are left.

Another way to further narrow this number is to remove interactions happening not at the same temporal field, meaning to remove cases where temporal target interval of vessel 1 $[t_1^1, t_{20}^1]$ does not intersect the target interval of second vessel $[t_1^2, t_{20}^2]$. After temporal filtering, the number reduces dramatically to 49401.

Lastly, spatial filtering is applied. It is done by taking both last points of predictor sequences and calculating the haversine distance between them. If the calculated distance goes above 30840 meters, case gets removed. Such number is chosen knowing that the maximum speed of cargo vessel is 771 meters per minute and the target sequence length is of 20 minutes, thus the calculation $771 \cdot 20 \cdot 2 = 30840$. Such spatial filtering leaves **7283** interactions between ships which are eligible for collision prediction.

4.2 Trajectory modelling

Trajectory modeling part of the research will come in two parts. In the first one, we will search for the most accurate model to predict trajectories. In the second part, we will use the model we found on 10 different shuffles of data to retrieve 10 different (noised) predictions which will later be used for collision predictions.

4.2.1 Modelling for accuracy

As stated in the methodology 3.3 part of this thesis we will use different hyperparameter variations on 4 different models. In the experiment, we will use different variations of layer sizes from 50 to 250 with a step of 50 and batch sizes of 2^n where n is from 4 to 7. Mean squared error is chosen as our loss function. We will employ ReduceLROnPlateau function which will decrease the learning rate by a factor of 0.1 starting from 0.001 to the minimum learning rate of 0.000001 to find the closest point to the minima.

The above strategy is used to retrieve such results:

Mean squared error $\times 10^5$ of ΔLatitude and ΔLongitude

Model architecture	Batch size	Cells				
		50	*100*	*150*	*200*	*250*
LSTM	16	6.4266	5.9247	5.6867	5.8211	5.4739
	32	6.3391	5.5768	5.8584	5.7155	**5.3705**
	64	7.2209	6.0076	5.9044	5.8244	5.7626
	128	7.2066	6.1720	5.9500	5.7380	5.6190
GRU	16	6.0010	5.8603	5.7251	5.4075	**5.3277**
	32	6.9350	5.7060	5.8688	5.8830	5.6949
	64	6.7323	6.4124	6.4705	6.9441	7.2444
	128	8.5512	8.1278	8.5675	7.1482	6.9194
Bi-LSTM	16	5.8454	5.4634	5.3939	**5.1251**	5.5904
	32	5.9555	5.7090	5.5515	5.4591	5.3967
	64	6.1895	5.8334	5.4724	5.6250	5.4865
	128	6.1605	5.8096	6.2893	5.6247	5.4579
LSTM AE	16	5.6324	5.4056	**5.3165**	**5.2201**	5.6930
	32	5.6934	5.4737	5.4559	5.6556	5.6089
	64	6.6791	5.9636	5.9093	5.6022	5.6785
	128	7.2050	6.2023	6.3952	6.1486	6.3171

In the table above are the results as mean squared error $\times 10^5$ of ΔLatitude and ΔLongitude, where green colored results correspond to the five best performing model results. Though useful in a general sense, mean squared error does not give interpretable results in the current medium as the data compared is both transformed to be differences in sequential coordinates. Besides the data training is done on standardized data which also is uninterpretable for the end user. To combat this a similar approach is applied that is found in paper [8] where mean absolute haversine error is calculated, this not only combats both of the problems mentioned previously but also is highly interpretable as the results correspond to average error distance in meters between two points:

Mean absolute haversine error in meters

Model architecture	Batch size	Cells				
		50	*100*	*150*	*200*	*250*
LSTM	16	235.36	218.28	212.95	212.98	202.03
	32	232.28	207.58	220.73	215.67	198.36
	64	261.98	229.85	223.61	223.12	219.55
	128	268.42	235.63	227.98	221.81	216.9
GRU	16	218.08	211.94	205.39	196.84	**194.62**
	32	243.37	207.14	210.24	212.34	205.91
	64	238.59	229.41	231.21	245.24	250.95
	128	261.22	266.15	273.92	252.21	247.77
Bi-LSTM	16	217.14	205.15	200.05	**192.51**	208.27
	32	224.98	214.89	210.15	205.63	202.29
	64	237.83	223.17	212.71	216.65	213.84
	128	233.19	223.34	239.24	216.12	212.99
LSTM AE	16	203.37	**196.29**	**192.54**	**187.78**	204.22
	32	208.23	198.12	196.71	202.62	201.75
	64	244.18	216.91	214.3	204.39	204.98
	128	259.16	229.62	235.84	230.44	232.01

Looking at the results presented as MAHE it becomes much clearer in general what the error is in meters. With a small margin, LSTM AE with 200 cells and a batch size of 16 gets the best results, thus it will be used further for trajectory prediction for collision prediction.

4.2.2 Modelling for collision prediction

To predict potential collisions a region-based approach is employed in this work. Regions are made by using noised data as a way to retrieve a distribution of predicted points rather than a single prediction.

To get different predictions the sequenced data is firstly split into a split of 85% and 15%, where the latter is for testing and is left unchanged. The remaining 85% is then split into 70% and 15% 10 times with a random state being changed each time, this way we end up with 10 datasets where the train and validation splits are always different and test split is left the same. This way we avoid data leakage.

On each of these distributions, 10 models, with the best model's hyperparameters from previous section 4.2.1, are trained to achieve 10 slightly different models. Applying these models to the test dataset we retrieve 10 different y_pred_test results.

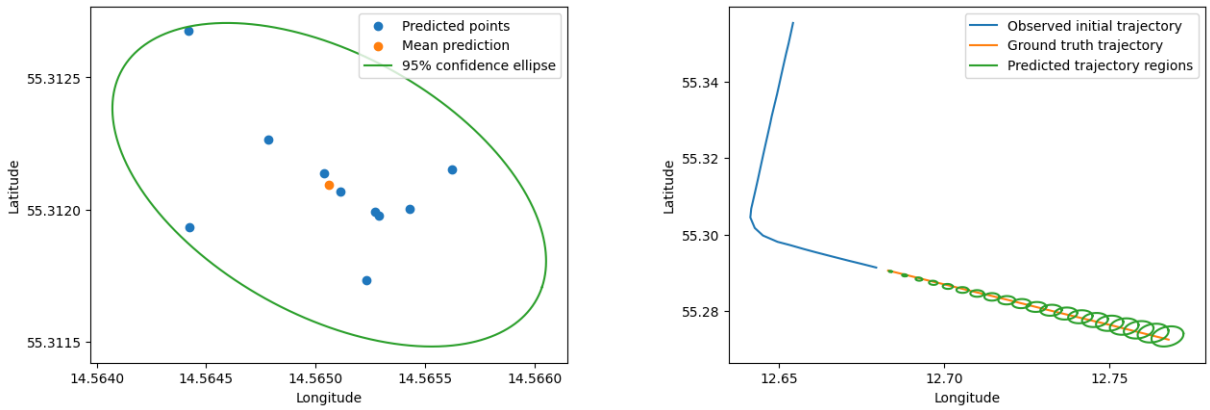
4.3 Collision prediction

Having 10 sets of predictions for the same test dataset allows us to create prediction regions that also could be considered as clusters. This allows us to develop methods that are used to calculate the likelihood of collision.

Performance of the methods will be compared by how well they can point out mock collisions as well as disregard cases where collision is improbable.

4.3.1 Ellipsoidal prediction region

To apply ellipsoidal prediction regions on our cluster we will use covariance matrix from each prediction. As ellipsoidal prediction regions are probabilistic measures, we need to apply a confidence measure. A 95% confidence level is chosen as it is a standard threshold widely accepted in scientific research.



(a) Example prediction region

(b) Example regions for predicted sequence

Figure 10: Ellipsoidal confidence prediction regions

Such regions will further be used in calculations of collision scores.

4.3.2 Ellipsoidal point intersection

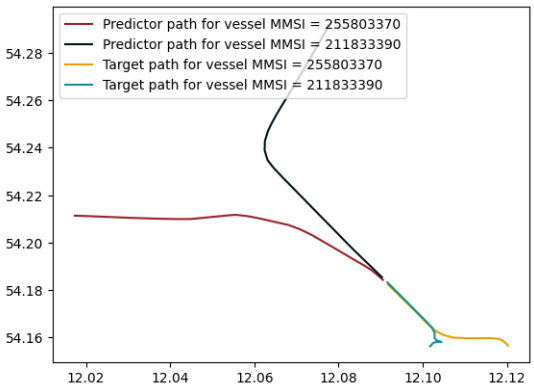
Using the ellipsoidal prediction regions built in previous section we firstly calculate number of sequences that had at least one intersection point in any of the predicted time points. This essentially puts the threshold of collision to 1 intersection point or in other words the intersection score to $\frac{1}{20}$. Result can be shown in a confusion matrix:

		Predicted	
		Yes	No
Actual	Yes	2	8
	No	2	7271

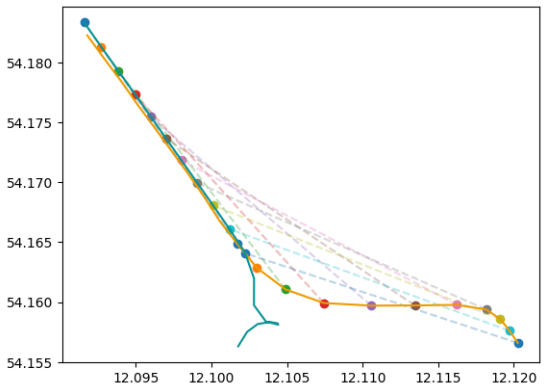
Unfortunately, only 4 sequence interactions were flagged as collisions. More critically, of these 4 only two are mocked collisions, thus giving only 2 true positives.

To better understand why such result occurred and whether it can be improved by changing the threshold, let's see the visual representations of the predicted collisions.

First interaction of ships that was flagged as a collision was between ships with MMSIs 211833390 and 255803370 at 2024-09-18 13:06:00. This occurrence is a false positive as the sequence was not a collision. To better understand why this case was flagged falsely we will analyze it further:



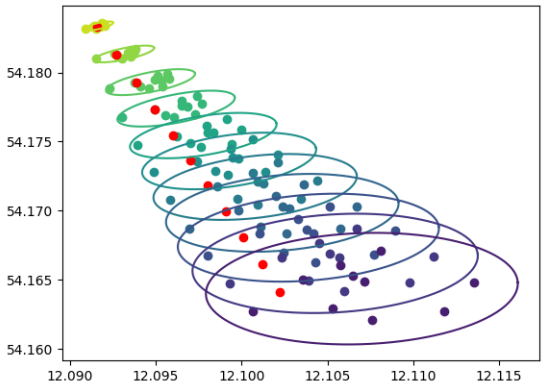
(a) Predictor and target paths of interaction



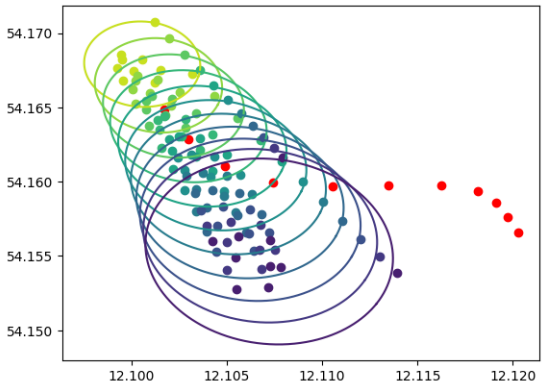
(b) Target path with time indicator lines

Figure 11: Original data plots of first interaction

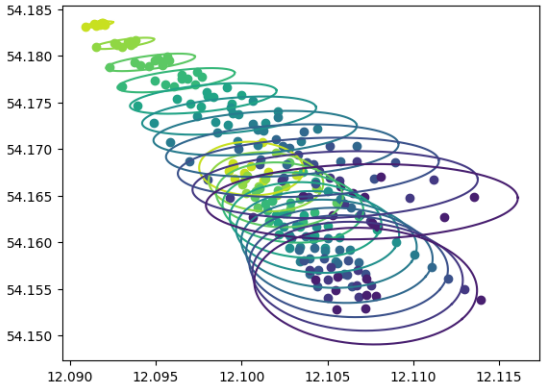
Though from the first plot 11a it could seem that the collision did happen, it would be an assumption with timeline not being taken into account. In picture 11b dashed line shows that there is distance between timepoints, thus the collision did not happen. To further analyze, let's look at the prediction regions.



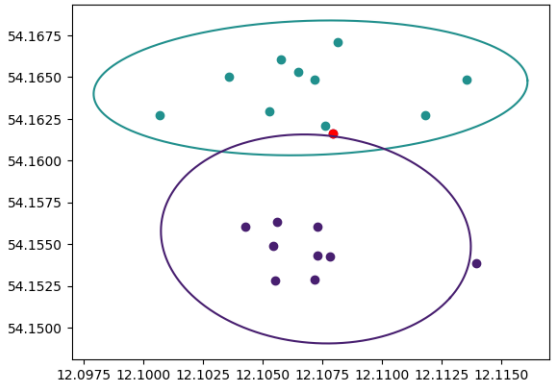
(a) First vessel's prediction regions and real target path



(b) Second vessel's prediction regions and real target path



(c) Both vessels' prediction regions

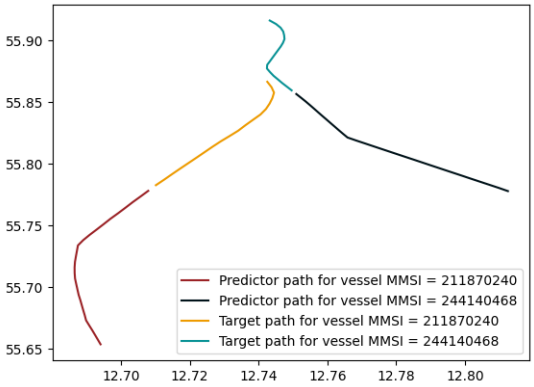


(d) Prediction regions of both vessels at the predicted collision timepoint

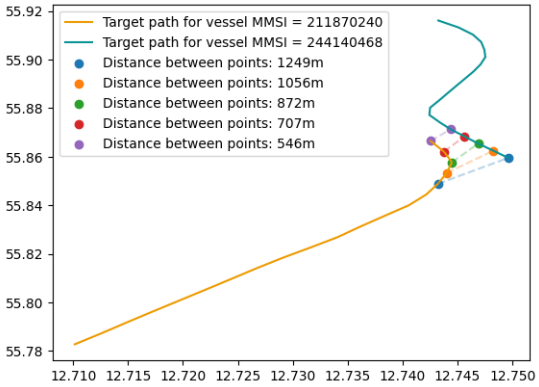
Figure 12: Both vessels' prediction regions with target path

By using figure 12 it is apparent that there are a few reasons why such false positive occurred. Firstly it is apparent that though first vessel's trajectory 12a was inside the predicted regions, second vessel's trajectory was mispredicted due to a sudden shift in movement as seen in figure 12b where red points indicate real target path. As seen in the figure 12d the intersection has only one point making the prediction score the lowest - 0.05. Such case, though incorrect, is not critical as the approach of two vessels was close.

Moving forward, another false positive happened in the interaction of vessels with MMSIs 211870240 and 244140468 at two time points - 2024-09-24 12:53:00 and 2024-09-24 12:54:00.



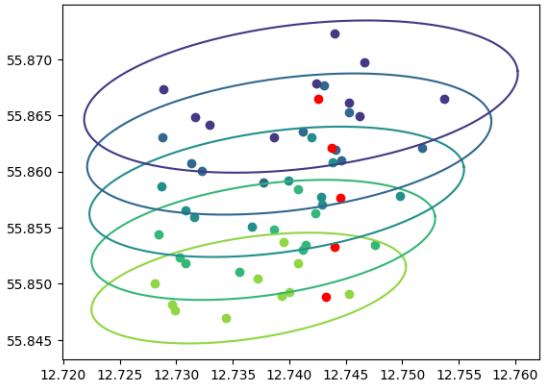
(a) Predictor and target paths of interaction



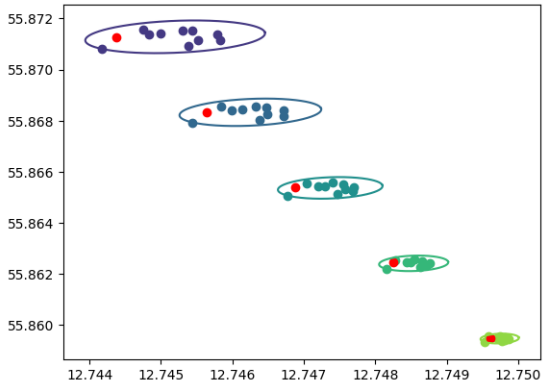
(b) Target path with time indicator lines

Figure 13: Original data plots of second interaction

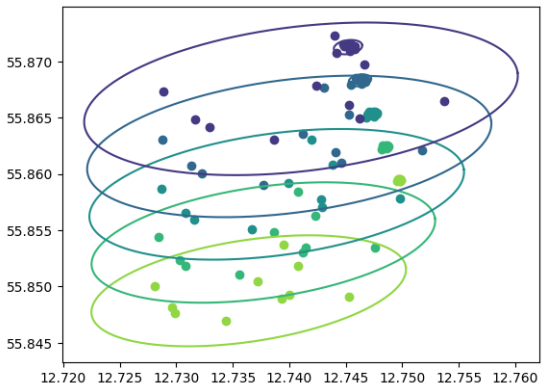
Similarly to the first interaction, this one was also identified as a possible collision, however in this case it is apparent from figure 13b that this interaction was near, at the closest point coming to 546 meters.



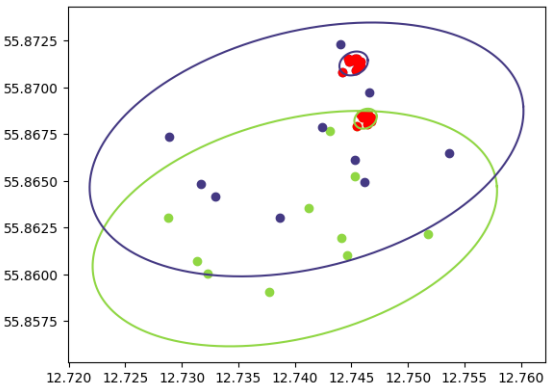
(a) First vessel's prediction regions and real target path



(b) Second vessel's prediction regions and real target path



(c) Both vessels' prediction regions

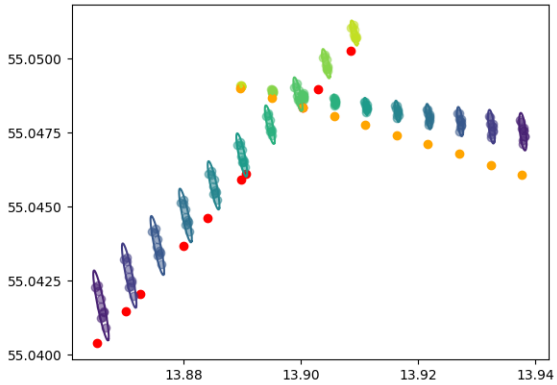


(d) Prediction regions of both vessels at the predicted collision time points

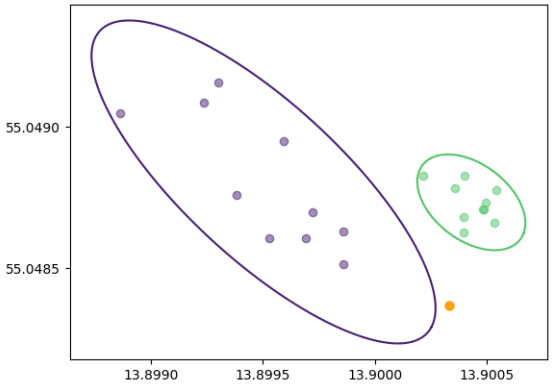
Figure 14: Both vessels' prediction regions with target path

As seen from both vessels' prediction regions plots 14a, 14b, they are indeed accurate. Such prediction regions result in two time points being predicted as collisions. Both of them have the prediction score of 0.5 as both time points have 10 intersection points. Lowering the threshold to >0.5 would discard these predictions however, as it is seen from 14d when two prediction regions of different sparsity collide, it is unlikely that the ones with bigger sparsity points will be included in the prediction, thus such case should stand.

Even though two false positives were identified, a much bigger problem is that only two collisions were identified as true. Thus further analysis of false negatives is required. Let's look at the prediction regions of these cases:



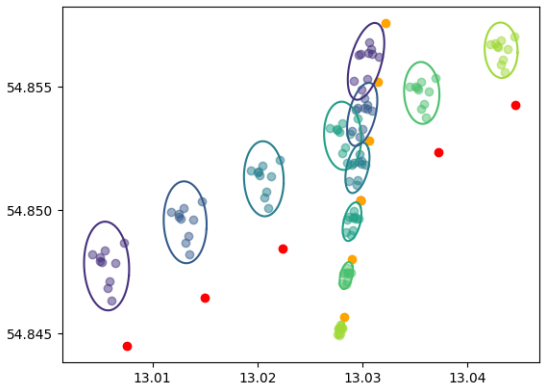
(a) Prediction regions of first false negative case with real target paths in red and orange



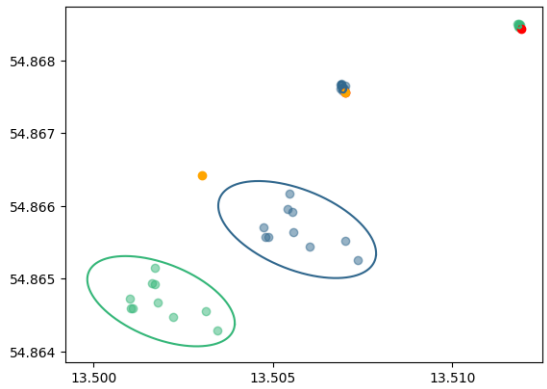
(b) First false negative's prediction regions at the time-point of collision

Figure 15: First false negative case

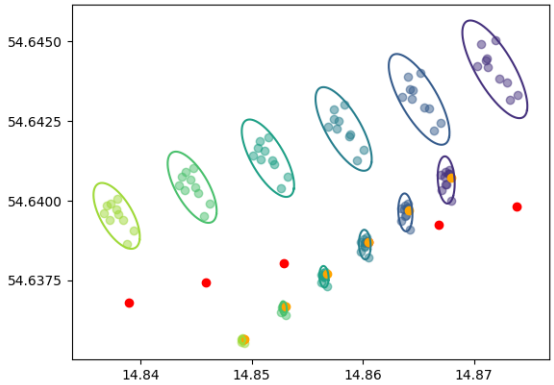
First false negative interaction happened due to two reasons, firstly the trajectory prediction was not accurate enough, as seen from the 15a, vessel's target path in orange shows that predicted regions are slightly off to the side, and vessel with the target path in red shows that speed of the vessel was predicted inaccurately. Second reason behind the false negative is the pass-by of the vessels as seen from figure 15b where orange indicates the real (mock) collision point of the vessels, such problem occurs as our predictions are done for every minute meaning that we are predicting regions every minute discretely rather than continuously. We will call this the pass-by problem. Similar problem also occurs with other false negatives:



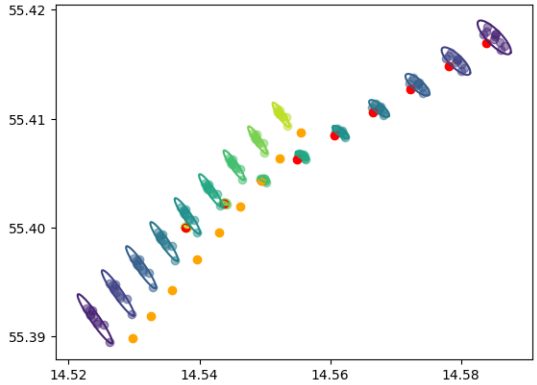
(a) Prediction regions of second false negative case with real target paths in red and orange



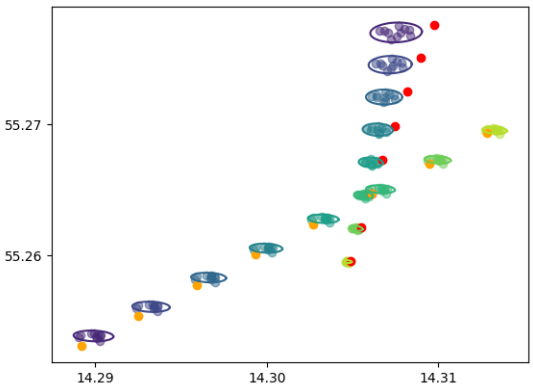
(b) Prediction regions of third false negative case with real target paths in red and orange



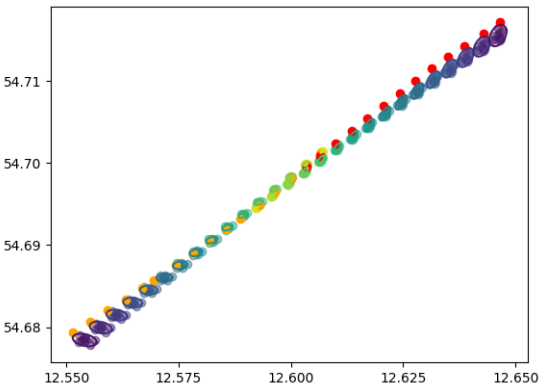
(c) Prediction regions of fourth false negative case with real target paths in red and orange



(d) Prediction regions of fifth false negative case with real target paths in red and orange



(e) Prediction regions of sixth false negative case with real target paths in red and orange

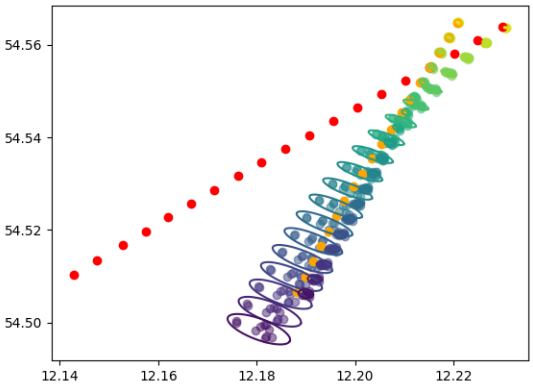


(f) Prediction regions of seventh false negative case with real target paths in red and orange

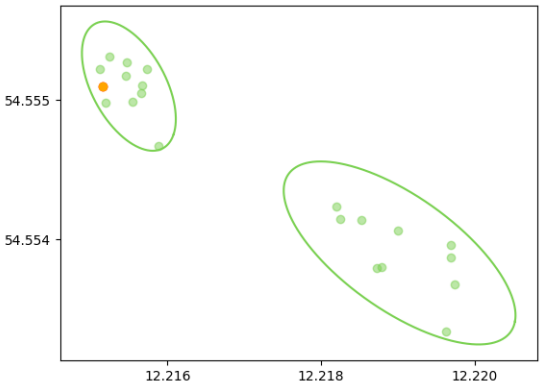
Figure 16: Other false negative cases with the pass-by problem

From these interactions, it is evident that this problem should be solved either by increasing the bounding ellipse or by using non-boundary based method.

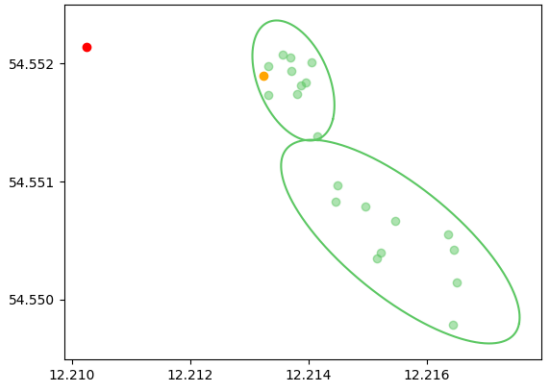
However, there is another problem that is exclusive to the ellipsoidal point intersection method.



(a) Prediction regions of eighth false negative case with real target paths in red and orange



(b) Eighth false negative's prediction regions at the time-point of collision



(c) Eighth false negative's prediction regions at one timepoint after collision

Figure 17: Eighth false negative case

Though the eighth false negative also suffers from the pass-by problem as seen from the figure 17b, figure 17c also shows the shortcoming of ellipsoidal point intersection method as two ellipses of the vessels collide it indicates a possibility of collision, however due to the prediction points being sparse, method is unable to identify this interaction as collision. One way to overcome this problem is to make more prediction points, however, such approach is highly computationally expensive and is unviable without high processing power. Another way to solve this problem is to consider areas of the bounding ellipses. Such approach is further explored in the next section.

To conclude, the ellipsoidal point intersection method suffers from its strict nature, giving a very low threshold for identifying collisions thus getting a very high false negative score. In the current medium, it is not as useful as we would like to get a more conservative approach, prioritizing safety over risk. Such approach could be more useful in a more confined environment where the prediction regions are naturally very close such as ports or natural narrowings of paths.

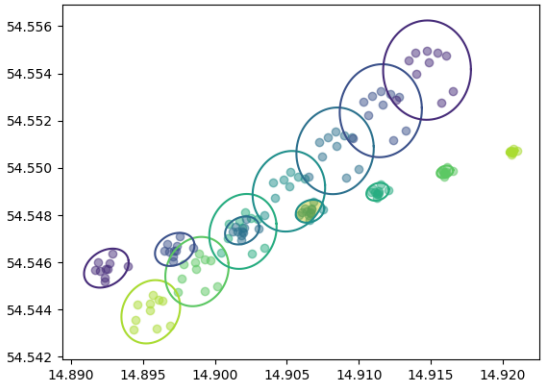
4.3.3 Ellipsoidal area intersection

To overcome the strictness of the point intersection method seen in figure 17c, area intersection can be applied. Similarly to the ellipsoidal point intersection method, we will use the lowest threshold, as area is a continuous metric the threshold will be any intersection that is >0. We will also calculate intersection over union.

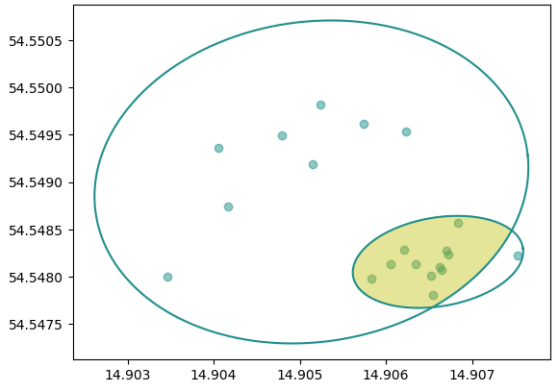
After using this method on all interactions the result can be shown in a confusion matrix

		Predicted	
		Yes	No
Actual	Yes	3	7
	No	3	7270

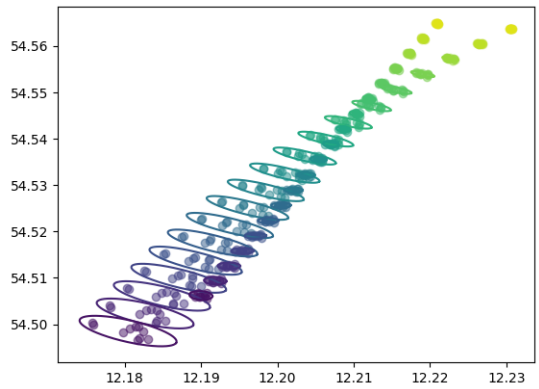
With a small improvement, the ellipsoidal area intersection method managed to flag 3 correct collisions:



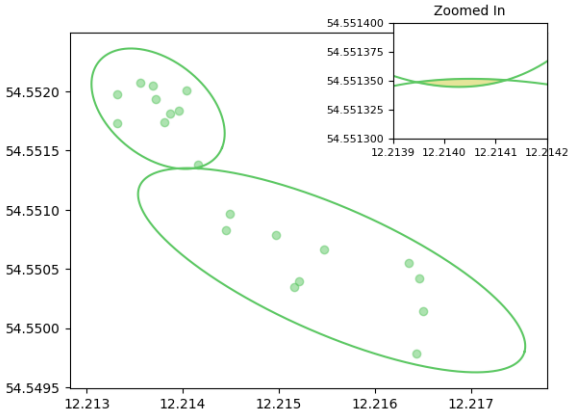
(a) Prediction regions of first true positive case



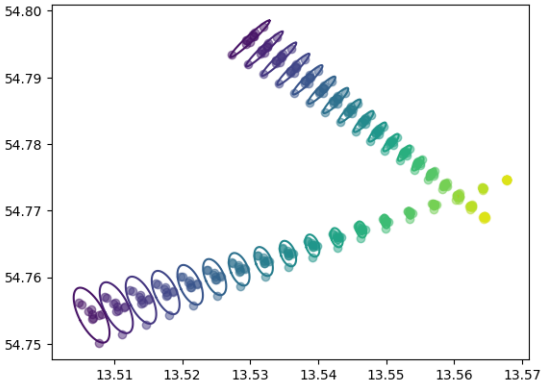
(b) Collision point of first true positive case - IoU = 0.09



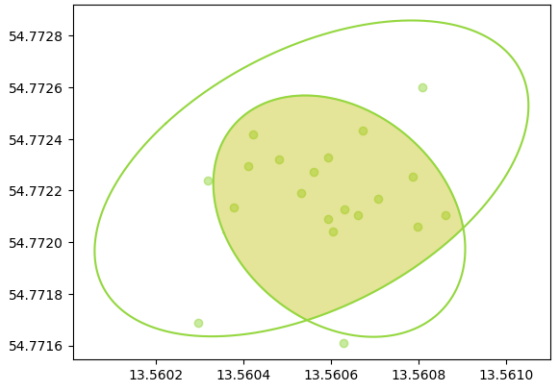
(c) Prediction regions of second true positive case



(d) Collision point of second true positive case - IoU = 0.0002



(e) Prediction regions of third true positive case

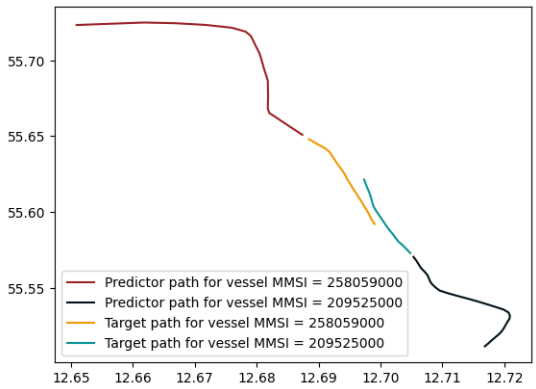


(f) Collision point of third true positive case - IoU = 0.38

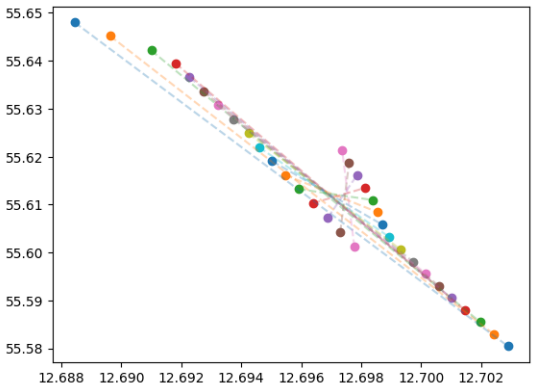
Even though the ellipsoidal area intersection method was able to identify one more collision event, lowering the threshold for the intersection area or IoU would not be helpful as the second predicted collision event was included due to the small threshold.

Looking at the false positives, two of the three predicted are the same from the ellipsoidal point intersection method as seen in figures 12, 14.

Third false positive can be analyzed further:



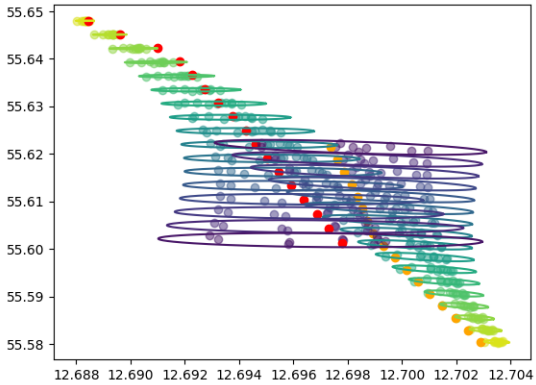
(a) Original predictor and target path of third false positive case



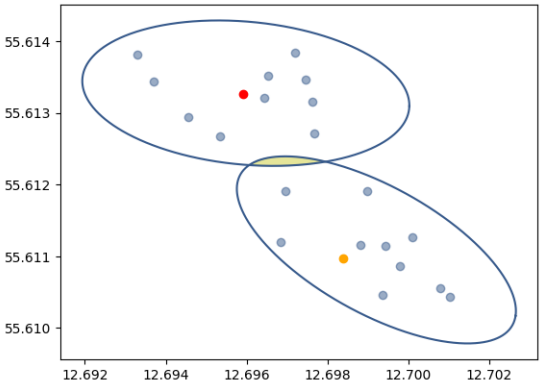
(b) Original target path of third false positive case with timepoint matching lines

Figure 19: Third false positive case's original paths

As seen from the figure 19, the vessels pass by at a close distance (closest being 298 meters), thus making this case a desirable false positive.



(a) Prediction regions of third false positive case with real target paths in red and orange



(b) Third false positive's prediction region at the point of predicted collision with IoU = 0.007

Figure 20: Third false positive case's prediction regions

Looking at the prediction regions of the third false positive case it is apparent that the case was predicted well, such cases should be identified, as well as mock collisions.

Other false negatives follow the same analysis as in section 4.3.2

Ellipsoidal area intersection, though solves the problem of predicted points not being inside the other ellipse when ellipses collide, still greatly suffers from the pass-by problem.

4.3.4 Z-score intersection

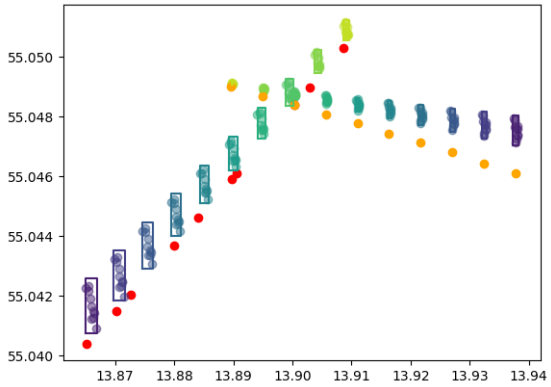
The idea behind the Z-scored bounding box area intersection method is analogous to the ellipsoidal area intersection method. Only difference is that instead of considering the confidence interval to be of 2 degrees freedom in the current section we will separate latitude and longitude variables to be of

one degree of freedom this way creating a bounding box. Firstly, Z-score = 2 is chosen, this means that around 95% of predicted latitude and longitude values independently will lie inside the according intervals.

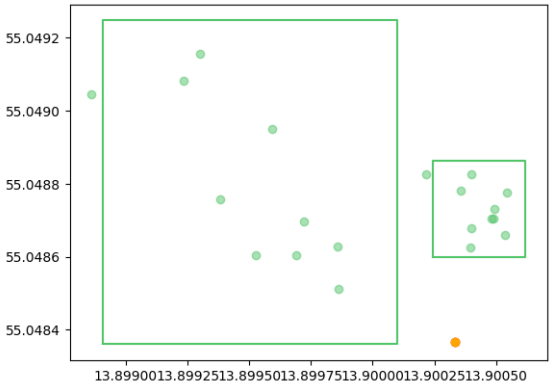
After doing the calculation we get the confusion matrix:

		Predicted	
		Yes	No
Actual	Yes	2	8
	No	3	7270

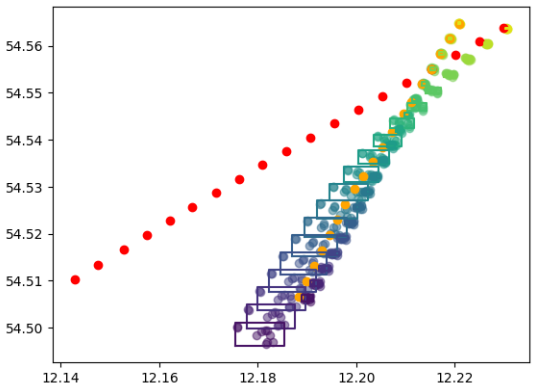
Bounding box with Z-score = 2 performs similarly to the ellipsoidal point intersection method however the result is still poor - only 2 collisions were identified. An analysis of false negatives is conducted:



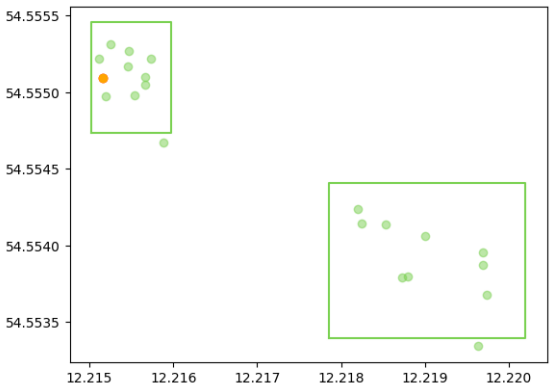
(a) Prediction regions of first false negative case with real target paths in red and orange



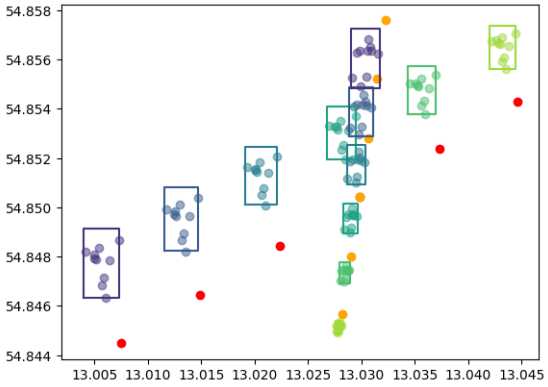
(b) First false negative's prediction region at the time-point of collision



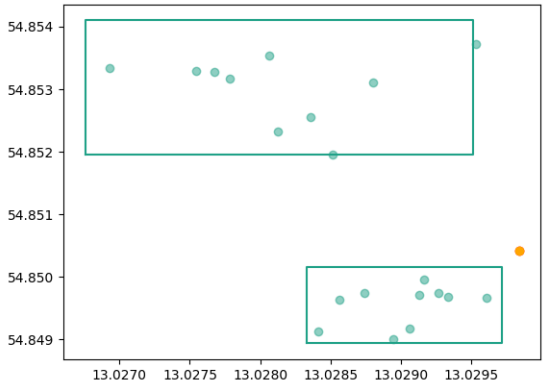
(c) Prediction regions of second false negative case with real target paths in red and orange



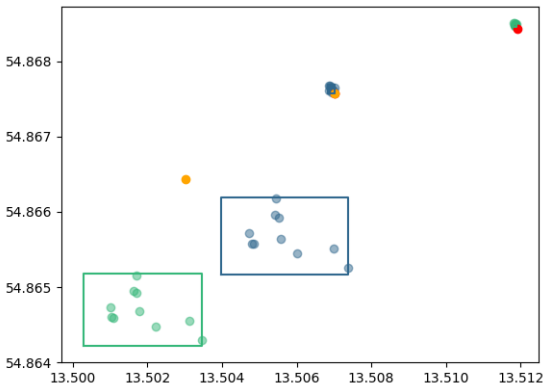
(d) Second false negative's prediction region at the time-point of collision



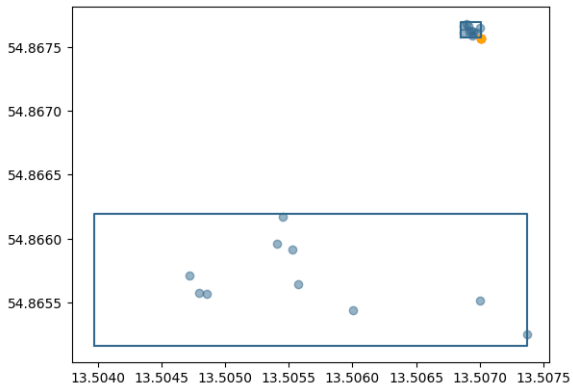
(e) Prediction regions of third false negative case with real target paths in red and orange



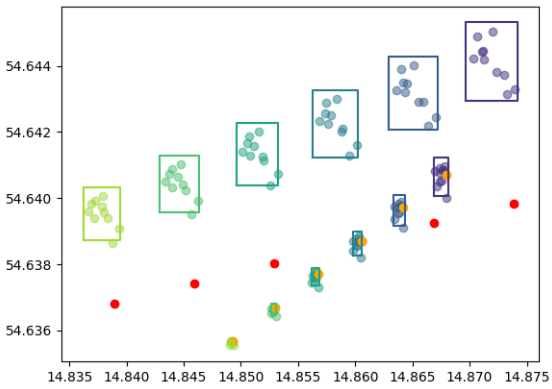
(f) Third false negative's prediction region at the time-point of collision



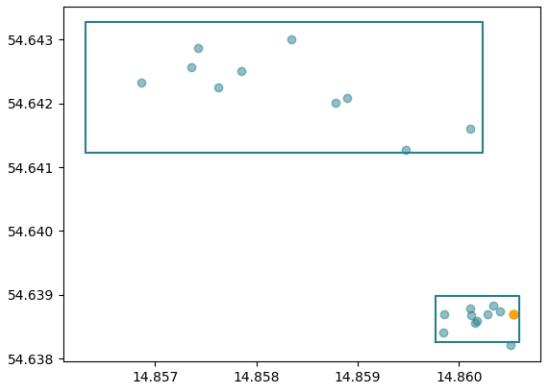
(g) Prediction regions of fourth false negative case with real target paths in red and orange



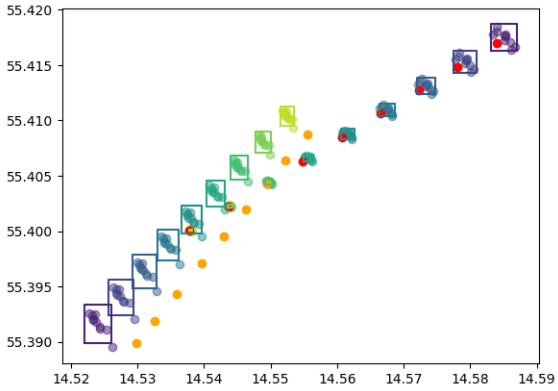
(h) Fourth false negative's prediction region at the time-point of collision



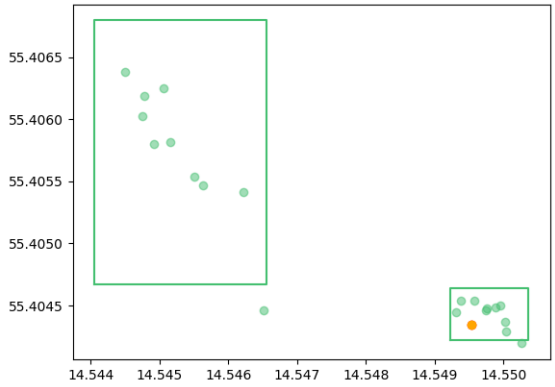
(i) Prediction regions of fifth false negative case with real target paths in red and orange



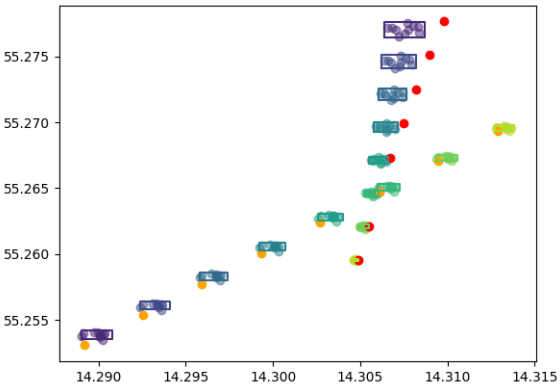
(j) Fifth false negative's prediction region at the time-point of collision



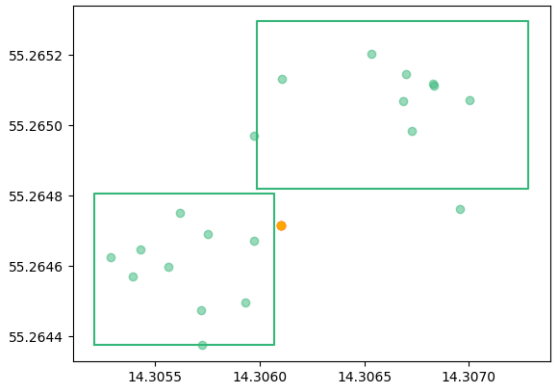
(k) Prediction regions of sixth false negative case with real target paths in red and orange



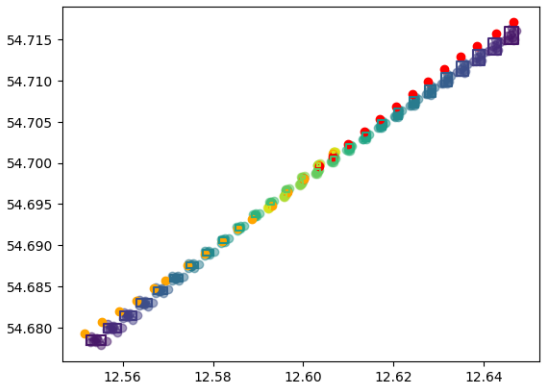
(l) Sixth false negative's prediction region at the timepoint of collision



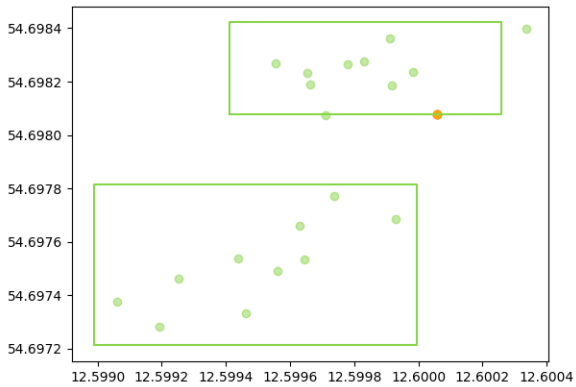
(m) Prediction regions of seventh false negative case with real target paths in red and orange



(n) Seventh false negative's prediction region at the timepoint of collision



(o) Prediction regions of eighth false negative case with real target paths in red and orange



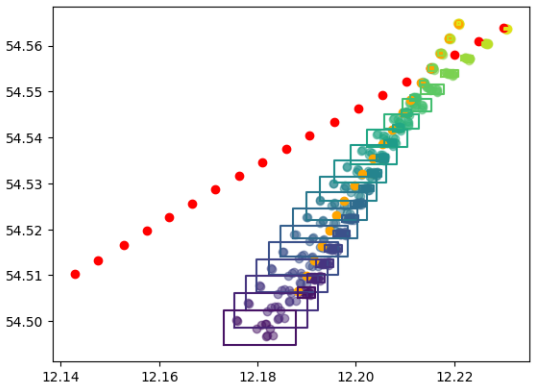
(p) Eighth false negative's prediction region at the timepoint of collision

Figure 21: Z-score = 2 bounding box prediction regions' false negatives

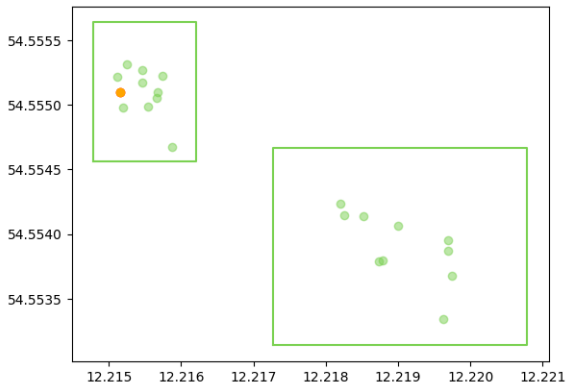
Evidently from figure 21, Z-score bounding box method encounters the pass-by problem, most prominently shown in figure 21n. To overcome it we will increase the Z-score to be 3. After such adjustment the confusion matrix results in:

		Predicted	
		Yes	No
Actual	Yes	5	5
	No	4	7269

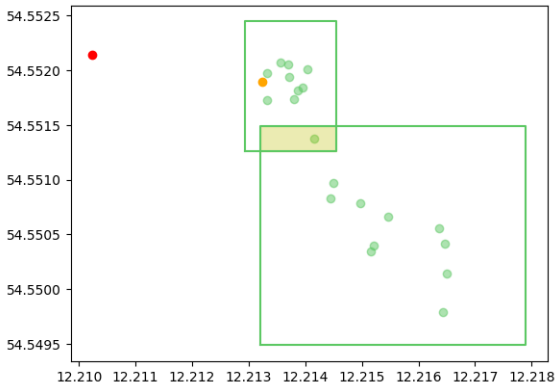
This is a much more desirable result. Let's further analyze cases that were re-evaluated as collisions after an increase in Z-score.



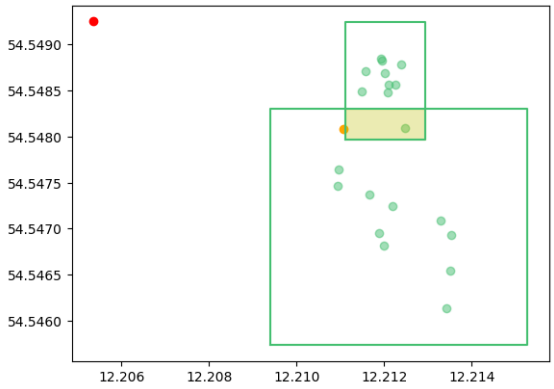
(a) Prediction regions of first new true positive case with real target paths in red and orange



(b) First new true positive prediction region at the time-point of collision



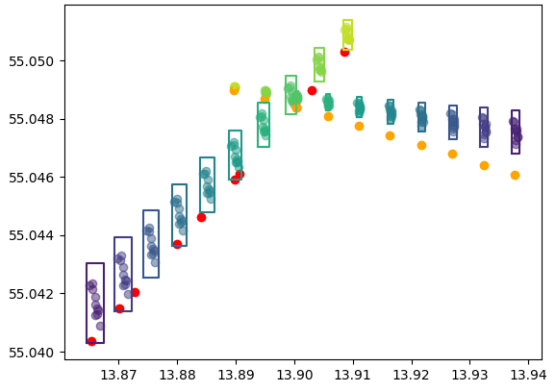
(c) First new true positive prediction region at one time-point after mock collision with IoU = 0.03



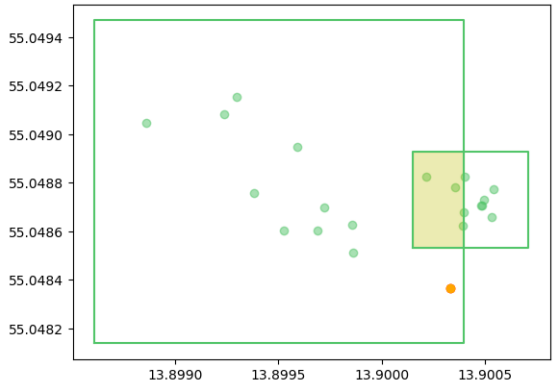
(d) First new true positive prediction region at two time points after mock collision with IoU = 0.04

Figure 22: First new true positive case

The case in the above figure 22 corresponds to the previously analyzed case 21c. Interestingly the prediction of collision is evaluated with a lead of one minute after the collision timepoint itself. Also, after the collision, 7 consecutive time points were evaluated as collision having IoU > 0.



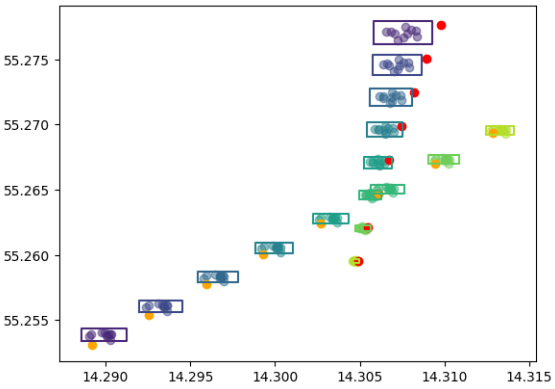
(a) Prediction regions of second new true positive case with real target paths in red and orange



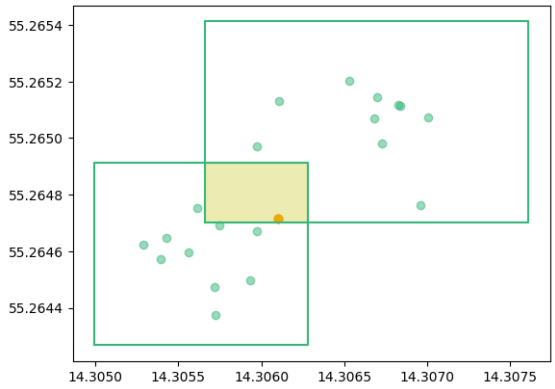
(b) Second new true positive prediction region at the timepoint of collision with IoU = 0.04

Figure 23: Second new true positive case

In the figure 23 another case is shown which directly corresponds to figure 21a. It is easily seen how increasing the Z-score makes the model more able to deal with the pass-by problem.



(a) Prediction regions of third new true positive case with real target paths in red and orange



(b) Third new true positive prediction region at the timepoint of collision with IoU = 0.06

Figure 24: Third new true positive case

Lastly, the final true positive case in figure 24 is compared to its prediction with Z-score = 2 in figure 21m. Similarly to the previously analyzed case, this one also overcomes the pass-by problem.

To conclude, the pass-by problem can be highly reduced by increasing Z-score bounding boxes, however, in some cases, the problem is still present and will require a more continuous approach.

4.3.5 Gaussian mixture model intersection

Another way to overcome the pass-by problem is to use the Gaussian mixture model. Such way not only helps by creating a likelihood approach and giving small but existing score on the extremes of predictions this way getting rid of confined boundaries but also making the predictions more prominent around the mean where the area/point intersection approach had the same importance in the edges of ellipses/bounding boxes as in the center.

Another difference to previously tried bounding box based methods is that the threshold can be calculated before setting it. So firstly we will calculate all of the GMM values (taking the highest per interaction).

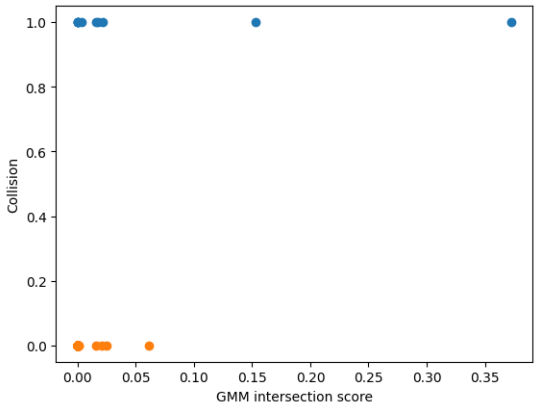


Figure 25: Collisions distribution against GMM intersection score

As seen from the figure 25, the distribution of interactions based on GMM intersection score majorly lie near 0, to be exact the smallest GMM of a collision is 3.86×10^{-16} below is the figure of such interaction.

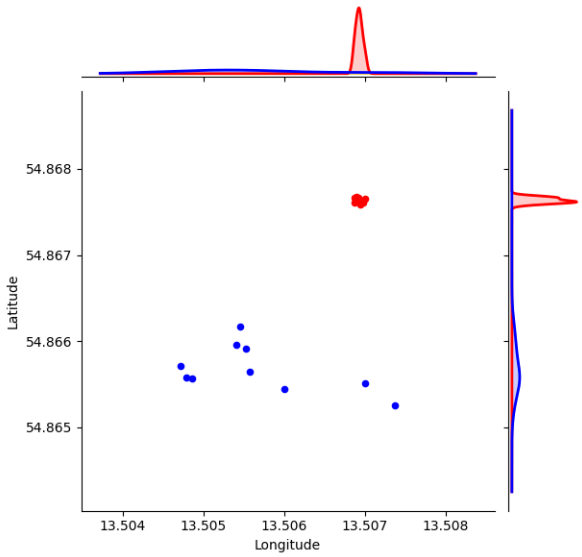


Figure 26: Lowest scored GMM prediction

This interaction can also be seen in figure 21h. The low score is a result of the predictions being early from one ship (thus a very narrow cluster) and also very late from the other vessel (thus most inaccurate both in direction and time). Setting this value as the threshold will result in the confusion matrix:

		Predicted	
		Yes	No
Actual	Yes	10	0
	No	41	7232

Considering the amount of data itself, the specificity in such case would be 99.4% which is highly optimal. However, for research's sake, we will test how lowering the threshold to the next lowest GMM score will affect the confusion matrix:

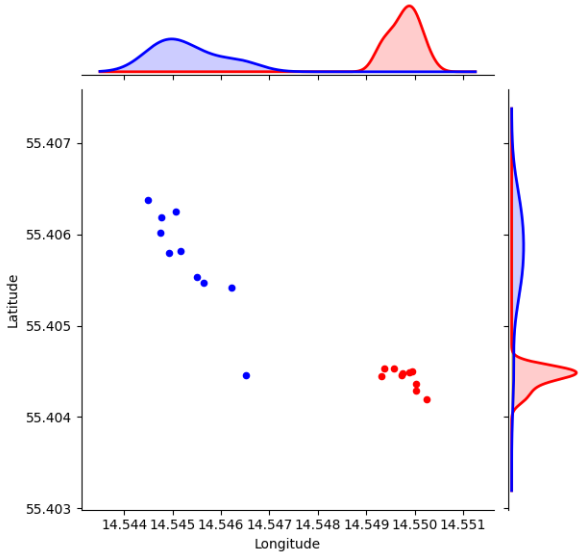


Figure 27: Second lowest scored GMM prediction

The above figure directly corresponds to the vessel interaction shown previously in figure 16d. Such case occurred due to incorrect direction prediction of one vessel. The GNN intersection score for this case is equal to 5.19×10^{-8}. Setting this value as the threshold will result in confusion matrix

		Predicted	
		Yes	No
Actual	Yes	9	1
	No	22	7251

Once again the result is highly favourable. A further link between thresholds (as GNN scores of mock collisions in ascending order) and false positives can be seen in the table:

Threshold	TP	FP	Specificity
3.86×10^{-16}	10	41	99.44%
5.19×10^{-8}	9	22	99.70%
2.66×10^{-7}	8	20	99.73%
1.09×10^{-5}	7	12	99.84%
3.30×10^{-3}	6	4	99.95%
1.56×10^{-2}	5	4	99.95%
1.73×10^{-2}	4	3	99.96%
2.16×10^{-2}	3	2	99.97%
1.53×10^{-1}	2	0	100.00%
3.72×10^{-1}	1	0	100.00%

Though useful in this particular example, generally such threshold setting approach has drawbacks as we are setting a threshold on seen data. To overcome this issue we can set the threshold to zero. This seems illogical as GMM is not discrete thus the values should never reach zero, however when calculating GMM scores we are mostly dealing with very low numbers and the minimum computational value for the float data type is 2.225×10^{-308}, any number lower than this is evaluated to be zero thus the threshold can be set to 0. After choosing such cut-off point we are left with a confusion matrix:

		Predicted	
		Yes	No
Actual	Yes	10	0
	No	396	6877

In this example, false positive cases increase however specificity is still maintained at 94.56% which is still objectively favorable.

Gaussian mixture model intersection score is a highly applicable method in such medium as it can easily overcome the pass-by problem which is highly prominent in bounding box based methods. Though the choice of the threshold could be made based on individual needs, in this case, the threshold corresponds to computational limits - the lowest float data type. This way, we are not only capturing all mock collisions but also keeping the specificity high enough. It is important to understand that though useful in such medium, such threshold would likely become useless if the ports would become the main area of interest as the interactions in these regions are happening in close proximity.

4.3.6 Silhouette value

Firstly, we will apply the silhouette score method to each interaction.

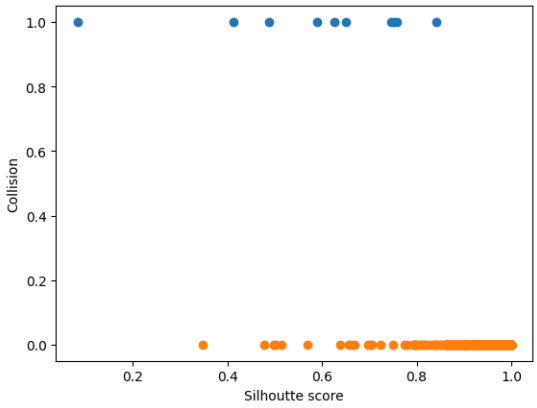


Figure 28: Collisions distribution against Silhouette scores

Looking at the distribution of collisions based on silhouette scores in figure 28, a similar result is visible as in reversed figure 25. As the silhouette score gets values between 0 and 1 based on how well or poorly two clusters are separated accordingly, the values should be reversed. Most basic reversal technique uses a function:

$$f(x) = 1 - x$$

After such transformation, the lowest value where the collision occurred is 0.15914904. Tuning the model to such value as the threshold results in the confusion matrix:

		Predicted	
		Yes	No
Actual	Yes	10	0
	No	30	7243

Resulting in 99.59 % specificity, which is even more precise result than seen in GMM. Similarly to the example set in the GMM section further link between thresholds and false positives is constructed.

Threshold	TP	FP	Specificity
0.15914904	10	30	99.52%
0.24218845	9	15	99.79%
0.24883867	8	15	99.79%
0.25425200	7	14	99.81%
0.35056688	6	7	99.90%
0.37334899	5	6	99.92%
0.41146003	4	6	99.92%
0.51294706	3	2	99.97%
0.5871981	2	1	99.99%
0.91700376	1	0	100.00%

Though with a slight deviation results from silhouette value based scores are also highly favorable having slightly better performance on the low threshold cases.

Although we have chosen the threshold to be the lowest silhouette score on a collision event, choosing the threshold on unseen data is quite a difficult task. To better understand the cut-off point which should be chosen, transformations of silhouette value can be used.

Before applying novel modified silhouette score, let's look at the reversed silhouette value kernel density function:

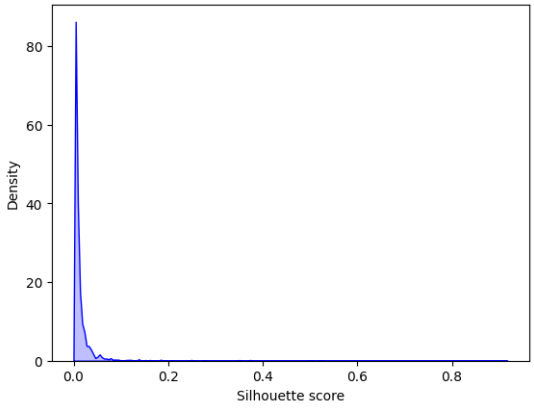


Figure 29: Silhouette value kernel density function

Without having any information on which cases are collision events, it is quite difficult to choose the threshold, though already there is separation visible at around 0.05. Putting silhouette value through the logistic function transformation results in density plot, where logistic parameter $n = 10$:

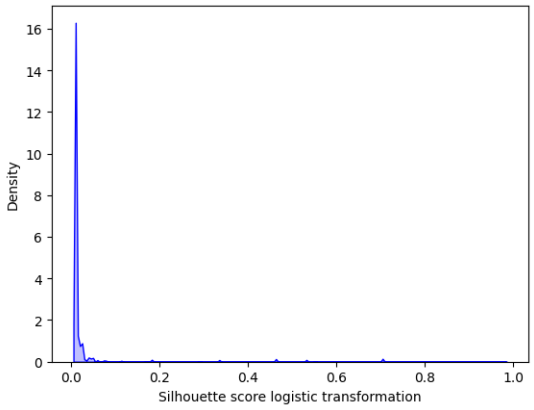
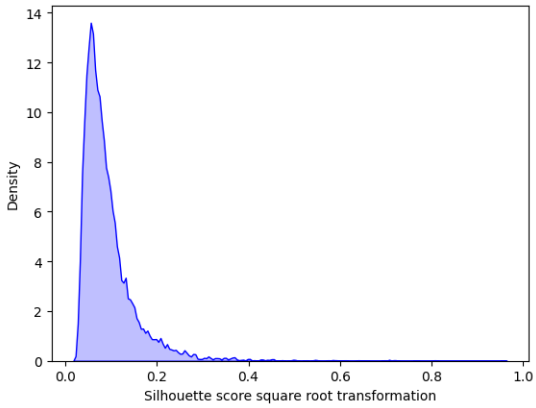


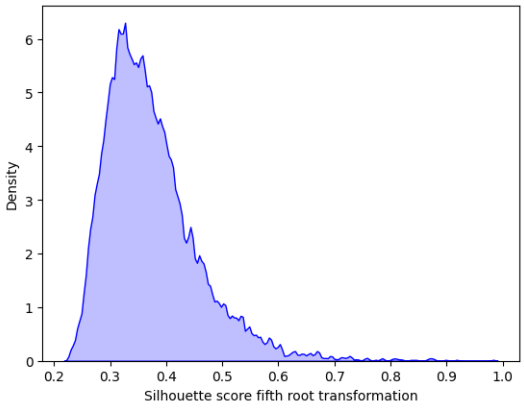
Figure 30: Logistic function transformation of silhouette score where logistic $n = 10$

As seen from the above figure not a lot of information has been unveiled. To better understand why, we can look at the logistic function figure 7b. It is evident that putting values that are already close to extremes (in this case where reversed silhouette score is close to 0) will put these values in the same extremes. Majority of such problem comes from the fact that the logistic function turning point is user-chosen, however, this is not ideal as the data is unique in each case thus making the logistic function not ideal in choosing the threshold.

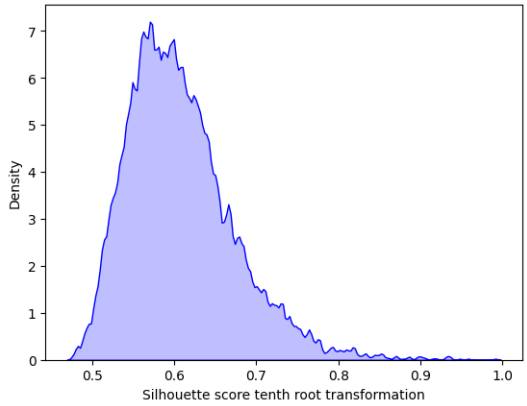
Another way to find the threshold is by using a variation of the root function.



(a) Silhouette value square root transformation



(b) Silhouette value fifth root transformation



(c) Silhouette value tenth root transformation

Figure 31: Silhouette value root transformations

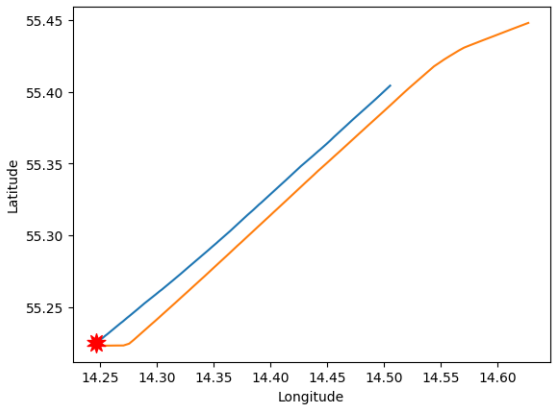
Such result is much more understandable as the values near 0 get distributed more evenly making it easier to visualize the possible cut-off point. Though in square root figure 31a it is not so apparent, in the fifth and tenth root figures it is visible that the threshold point should be around 0.6 and 0.8 accordingly, which converts to 0.08 and 0.11 accordingly, which though being lower threshold than previously chosen 0.15914904, makes the prediction more conservative which is the aim for this study.

To conclude, the modified silhouette score can be used as a high precision method to identify possible collision events. However, to better understand which threshold should be used, more true events (mock collisions) should be generated, in such case a comprehensive ROC (Receiver operating characteristic) curve could be examined to receive a more general threshold value.

5 Real collision analysis

As marine vessel collisions are extremely uncommon events, collision prediction is a highly challenging task as collision data is scarce. To test the theory to some extent, mock collisions can be used as done in section 4.1.1, however, it will never be the same as in a real example as there are no human factors in the mock collision. However, one collision example exists in the database. The collision event

happened on the 13th of December, 2021, the vessels involved had MMSIs: 219021240 and 232018267. The collision caused the capsize of one vessel where one of two crew members was found dead [22]. Visually this event looked like this:



(a) Collision on map

(b) Collision aftermath - capsized vessel, picture from article by Independent [7]

Figure 32: Collision event visualized

Though catastrophic, this event yields significant data for testing collision prediction methods. To have a comprehensive analysis of the collision, we will test all of the methods proposed in this study. Besides we will test 3 differently timed sequences, i.e. predictor variable will be taken 3 times for each vessel, firstly we will see how the methods perform when the predictor sequence ends 20 minutes before collision, this way we will see whether the collision could have been avoided at the earliest possible time. Another test will be conducted where the predictor sequence ends 10 minutes before the collision and lastly, where the predictor sequence ends 3 minutes before the collision. Such intervals have been specifically chosen so that the sequence with 10 minutes left until collision should predict the collision around the middle of the target variable and the predictor sequence with 3 minutes left was chosen as that is the exact time point where the vessel with MMSI 232018267 made an anomalous turn towards the other vessel.

5.1 Predictor sequence ending 20 minutes before collision

To better understand the results, firstly we will visualize the predictor sequence:

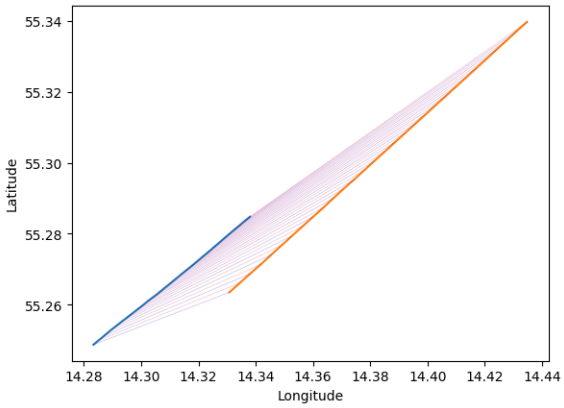


Figure 33: Predictor sequence 20 minutes before collision

In the figure 33 timepoint specific lines were added to better understand the behaviour of both vessels. It is apparent that vessel 232018267 was moving quicker than vessel 219021240. At the beginning of this sequence, the distance between vessels was 8.64km, and at the end of the sequence - 3.43km.

To further analyze, this sequence, we will apply methods used in section 4.3

Firstly, ellipsoidal methods are applied:

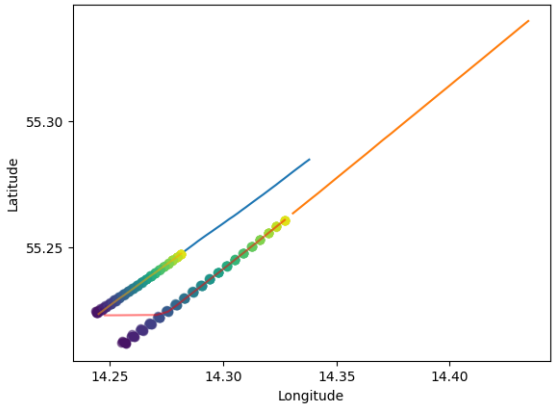
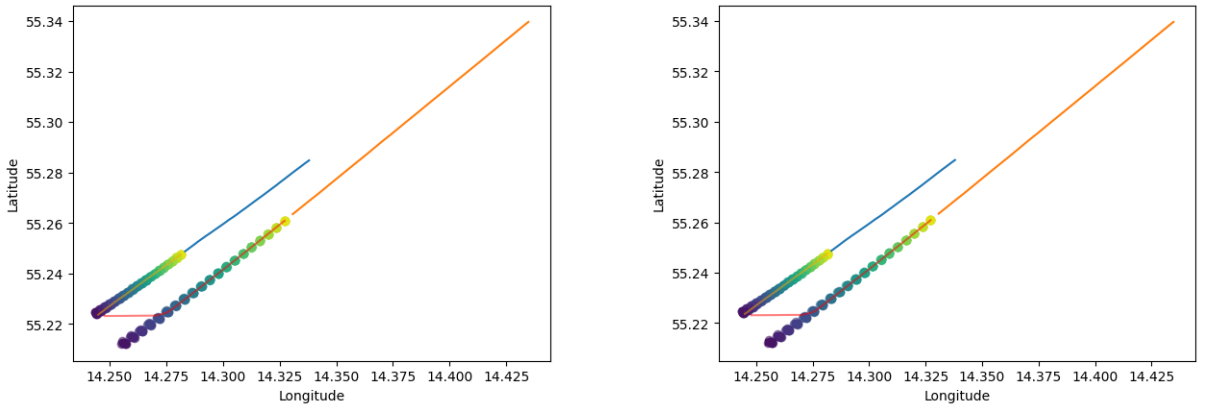


Figure 34: Predicted sequence using ellipsoidal prediction regions

As apparent from the above figure, both - ellipsoidal point intersection and area intersection methods are ineffective in this case, mainly it is due to the reason that vessels are moving in parallel, besides the movement of predictor sequences is quite monotonous thus the prediction regions are narrow.

Using Z-score bounding box intersection methods yields similar results:



(a) Predicted sequence using Z-score bounding box where Z-score = 2

(b) Predicted sequence using Z-score bounding box where Z-score = 3

Figure 35: Z-score bounding box prediction regions

Clearly, bounded methods work poorly on such example as the predicted sequences are parallel. To combat this, we will apply the Gaussian mixture model intersection method:

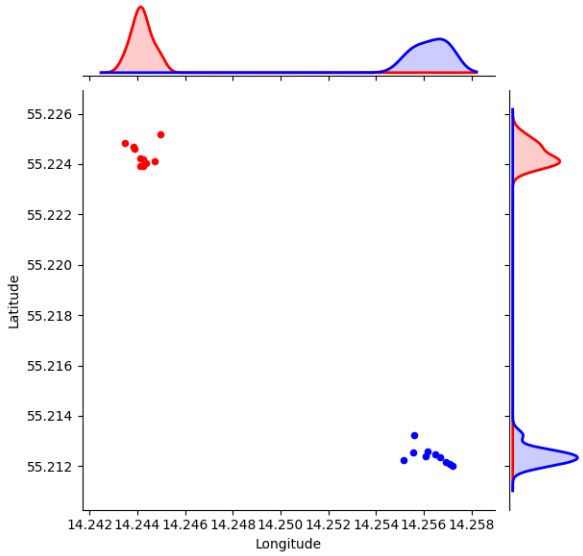


Figure 36: Predicted point with highest GMM intersection area 4.67×10^{-155}

Though the result of intersection being 4.67×10^{-155} is low, it is clear that it being non-zero becomes identifiable, as the threshold for the GMM intersection method is 0.

Further, the silhouette score based method is applied. At the timepoint of collision reversed silhouette value reaches 0.05, which transformed to a square, fifth, and tenth roots corresponding to 0.22, 0.55, and 0.74 accordingly. Unfortunately, these values fall just under the cut-off points selected in section 4.3.6.

These findings indicate that in this particular example, only the GMM intersection method was able to identify possible collision

5.2 Predictor sequence ending 10 minutes before collision

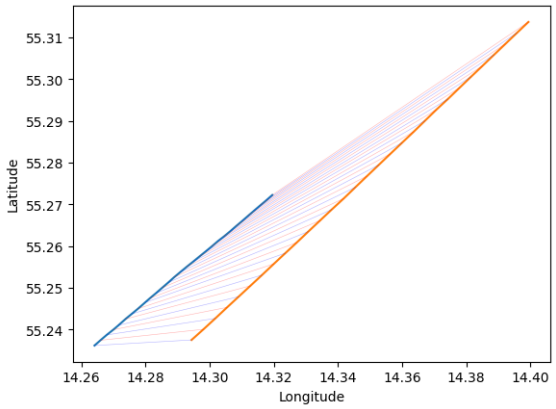
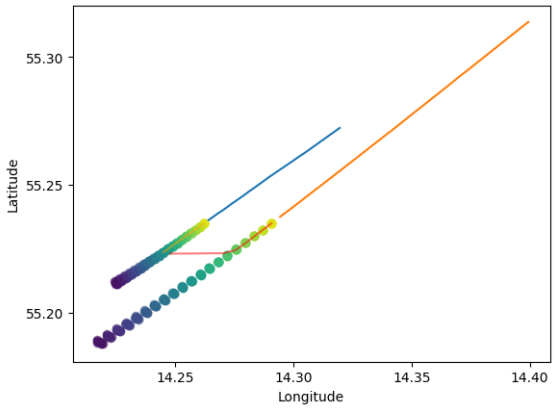


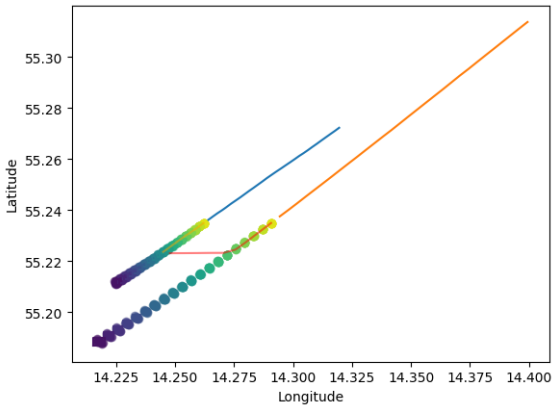
Figure 37: Predictor sequence 20 minutes before collision

Example shown in figure 37 is similar to the predictor sequence ending 20 minutes before collision in the sense that the heading of both vessels is similar, however at the last timepoint of the predictor sequence, vessels are much closer to each other, distance between them - 1.92km.

Due to vessels' directions being still parallel, the results from the bounded methods are still the same - no predicted collision:



(a) Predicted sequence using ellipsoidal prediction regions



(b) Predicted sequence using Z-score bounding box where Z-score = 3

Figure 38: Predicted sequences using bounded prediction regions

Further, the GMM intersection area is calculated:

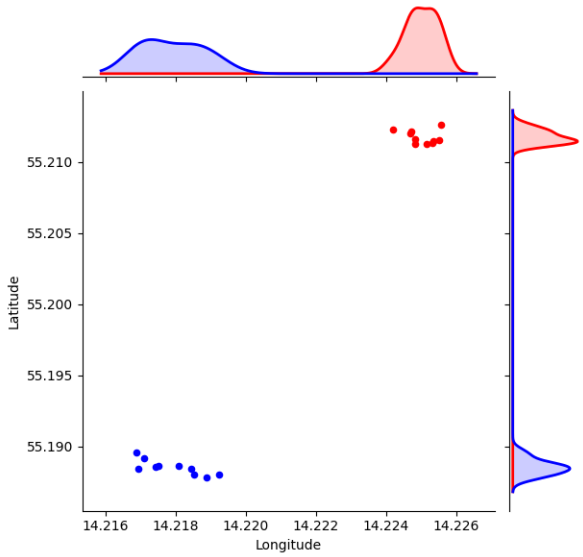


Figure 39: Predicted point with highest GMM intersection area 1.697×10^{-266}

Interestingly, the closest predicted point is modeled at the timepoint, 10 minutes later than the collision event, however, this is logical as the spread of predicted regions helps GMM in capturing event the smallest density functions' intersection area.

Lastly, the highest reversed silhouette value of the sequence is 0.04 which after transformations will also be lower than the threshold set in section 4.3.6.

Such results conclude that GMM was, again, the only method able to identify the collision while there was no anomalous behaviour in any of the vessels.

5.3 Predictor sequence ending 3 minutes before collision

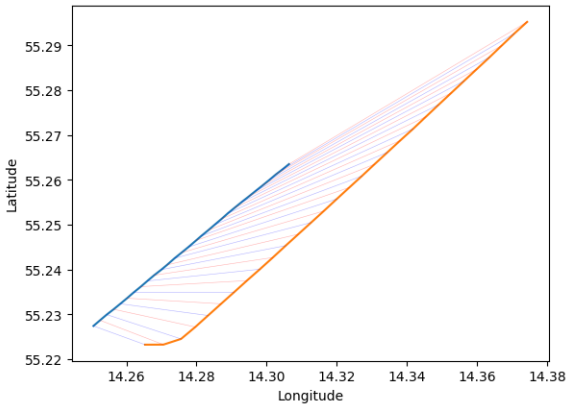
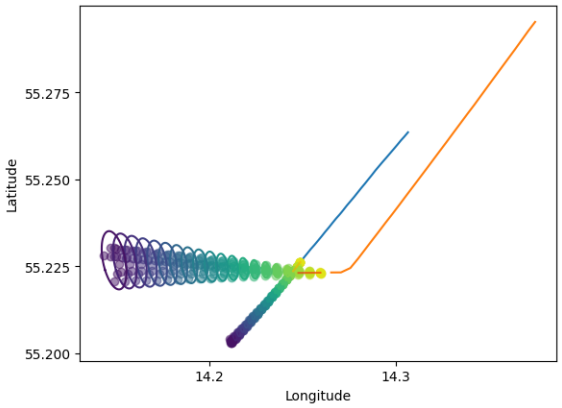
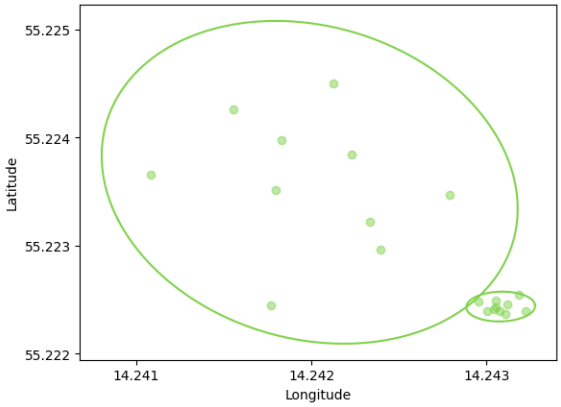


Figure 40: Predictor sequence 3 minutes before collision

Differently from cases with 10 and 20 minutes left until collision, this predictor sequence includes the anomalous turn of vessel 232018267. The distance between vessels at the closest time point - 1.04km.



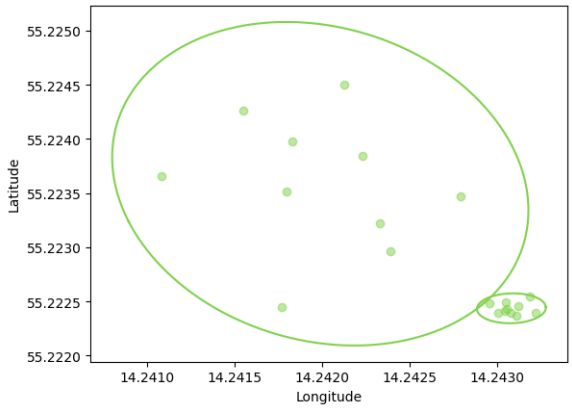
(a) Predicted sequence using ellipsoidal prediction regions



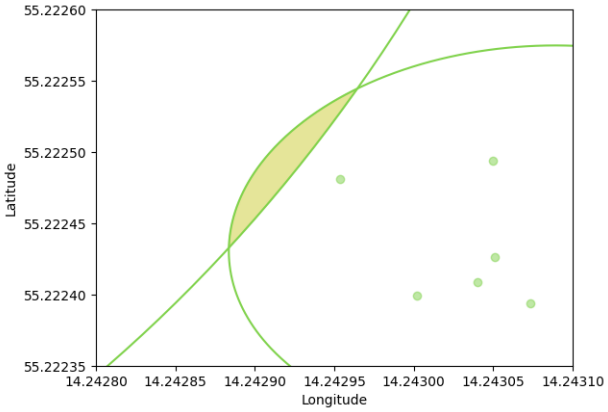
(b) Prediction regions at the timepoint of collision

Figure 41: Predicted sequences using ellipsoidal point intersection method

Though from first figure 41a it may seem that ellipsoidal point intersection method should be able to identify the collision, figure 41b shows the same issue that happened in figure 17 where though ellipses collide, points inside them do not intersect each others' ellipse. Thus method of ellipsoidal point intersection becomes useless in this example.



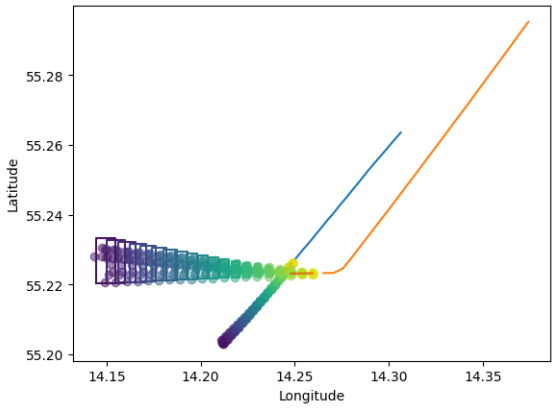
(a) Prediction regions at the timepoint of collision with IoU = 0.03



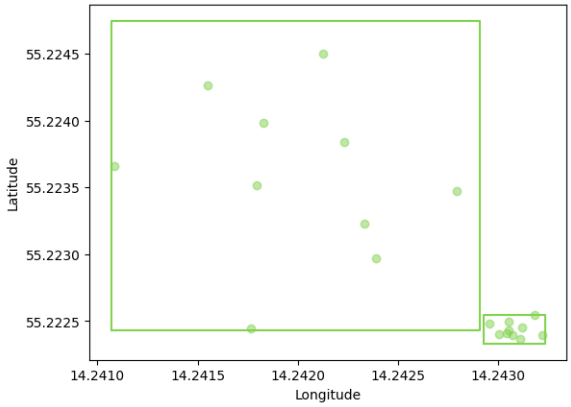
(b) Close up of prediction regions area intersection

Figure 42: Predicted sequences using ellipsoidal area intersection method

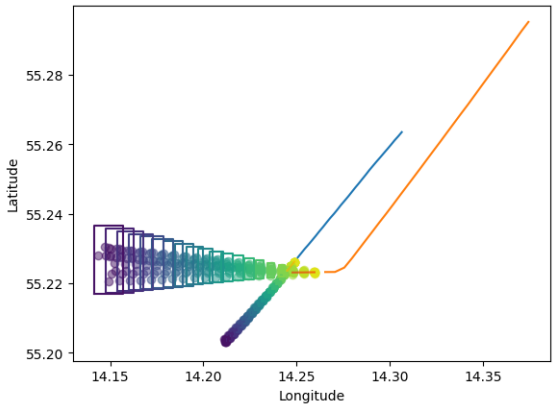
Using the ellipsoidal area intersection method, we are able to identify the collision event. Though it is able to overcome the shortcoming of the ellipsoidal point intersection method, the prediction is very close to being the pass-by problem, thus other methods should be tried as well.



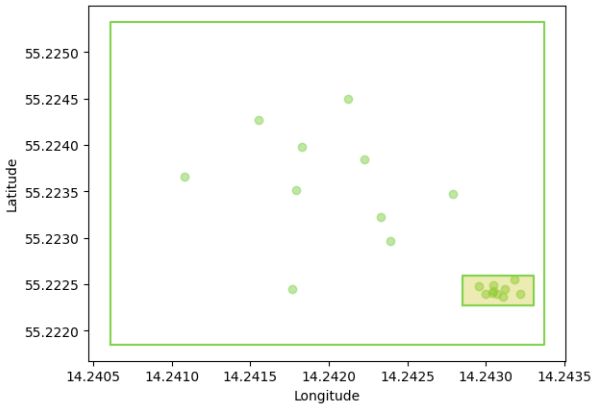
(a) Predicted sequence using Z-score bounding box intersection method with Z-score = 2



(b) Z-score bounding box region with Z-score = 2



(c) Predicted sequence using Z-score bounding box intersection method with Z-score = 3



(d) Z-score bounding box region with Z-score = 3, IoU = 0.02

Figure 43: Collision prediction using Z-score bounding box intersection method with Z-scores 2 and 3

Clearly from the figure 43, when using Z-score = 2, bounding boxes pass by each other however increasing the Z-score to 3, deals with this problem swiftly, getting the IoU result of 0.02.

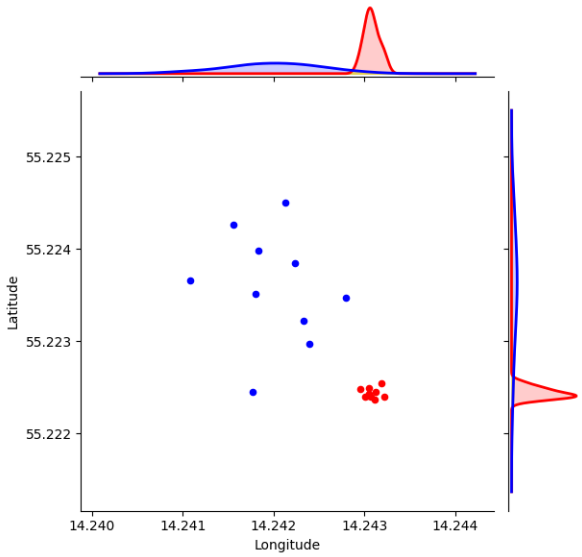


Figure 44: Predicted point with highest GMM intersection area 0.003

Once again, such GMM results correspond to a high likelihood of collision. Thus making this method highly applicable in the medium.

Lastly, after calculating the reversed silhouette value we get 0.38 which after transformation of square, fifth, and tenth root equates to 0.61, 0.82, and 0.91. These results clearly go over the threshold, i.e. equating to predicted collision.

6 Results

In this thesis, multiple trajectory prediction and collision risk assessment methods were applied.

To predict further trajectory of a vessel 4 RNN based models were compared (LSTM, GRU, Bidirectional LSTM, and LSTM autoencoder) in varying cell and batch sizes from 50 to 250 and from 16 to 128 accordingly. Using a comprehensive - mean absolute haversine error loss parameter LSTM AE with 200 cells and batch size of 16 with a small margin showed to be superior to other models receiving mean absolute haversine error of **187.78** thus it was chosen as trajectory predicting model.

It was then used on 10 shuffles of train and validation splits to receive 10 slightly noised variants of prediction which were used to form trajectory prediction regions.

Methods	**Sensitivity**	**Specificity**
Ellipsoidal point intersection	20%	99.97%
Ellipsoidal area intersection	30%	99.96%
Z = 2 bounding box area intersection	20%	99.96%
Z = 3 bounding box area intersection	50%	99.95%
Gaussian mixture model intersection	100%	94.56%
Modified silhouette score	100%	99.59%

Table 1: Collision risk assessment using sensitivity and specificity values

For collision risk assessment, 5 different approaches were used (with two modifications of the Z-score

bounding box area intersection method) and evaluated using two metrics - sensitivity and specificity as seen in table 1. Two collision risk assessment methods are based on ellipsoidal prediction regions (point intersection and area intersection). For the ellipsoidal point intersection method, only 2 mock collisions were correctly identified out of 10, giving a sensitivity score of 20% and specificity score of 99.97%.

Similarly, using the ellipsoidal area intersection method the result is below par giving 3 out of 10 correctly identified mock collisions. Thus sensitivity in this example is 30%. Specificity for this method is 99.96%.

Such poor results mainly come from the bounded nature of these methods. Besides these methods suffer from the pass-by problem where two subsequent ellipsoidal prediction regions do not intersect even though true target paths collide.

To mitigate this problem Z-score based bounding box area approach was used. Using Z = 2 resulted in similar results as ellipsoidal methods evaluating sensitivity to 20% and specificity to 99.96%. However, increasing the Z-score value to 3, improved the results to 50% sensitivity and 99.95% specificity. Nevertheless, the results in boundary based methods proved to be poor overall.

To overcome such poor performance, a non-boundary based approach was utilized starting with Gaussian mixture model intersection. Setting the threshold to the minimum computational float number resulted in 100% sensitivity meaning that all 10 mock collision cases were identified. The specificity for such method was lower than previous approaches - 94.56%. However, such result is desirable as the false positive cases (where collision did not happen but it was predicted as one) were close proximity events.

Last collision prediction method used was a novel custom-developed silhouette value based score which was transformed using square, fifth, and tenth roots. For this method, the smallest score of collision was chosen as the threshold, this way all collisions were evaluated correctly having 100% sensitivity and 99.59% specificity.

Lastly, all methods were applied to a real-case collision. The collision predictor variable was made into 3 different sequences where the sequence ended 20, 10, and 3 minutes before the collision.

In line with predictions, the ellipsoidal point intersection method was unable to identify the collision even when the predictor sequence ended 3 minutes before the collision. Such predictor sequence reveals the anomalous turn of one vessel that resulted in the collision as seen in figure 41

With a small margin, the ellipsoidal area intersection method was able to identify the collision, however, it was only done with 3 minutes remaining to the collision.

Z-score based method was only able to identify the collision when Z-score = 3 and when the predictor sequence ended 3 minutes before the collision.

GMM predicted possible collision on all 3 predictor sequences, retrieving maximum GMM intersection scores of 4.67×10^{-155}, 1.697×10^{-266} and 0.003 for the corresponding 20, 10, and 3 remaining minute predictor sequences. Such results especially prove GMM to be superior in the sequences where 20 and 10 minutes are remaining.

Lastly, the author's developed custom silhouette value based method was used. Though it managed to clearly predict the collision in the predictor sequence with 3 minutes remaining it was unable to succeed in the predictor sequences with 20 and 10 minutes remaining. Mainly such an issue occurred

due to a non-generalized threshold.

To conclude, results proved to favor the performance of the Gaussian mixture model as its continuous nature managed to capture even the smallest area intersection, which proved to be useful in the real-case analysis.

Conclusions

In this thesis, the research objectives outlined in the introduction were addressed, as detailed below:

1. The reviewed literature though gave an important background for trajectory prediction using variations of recurrent neural networks, more importantly, it showed a lack of research done in the evaluation of boundary and non-boundary based collision risk assessment methods in open-sea scenarios.

2. The theoretical foundation was established for four trajectory prediction methods: LSTM, LSTM Autoencoder, Bi-directional LSTM, and Gated Recurrent Unit. Additionally, the paper provides a theoretical explanation for five collision prediction methods: Ellipsoidal Point Intersection, Ellipsoidal Area Intersection, Z-Score Bounding Box Intersection Area, Gaussian Mixture Model intersection, and the author's novel modified Silhouette score method. These methods are subsequently compared in the experimental section of the paper.

3. The methods theoretically described in the methodology section of the thesis were applied in the experiment section to perform trajectory and collision prediction. The LSTM Autoencoder with 200 cells and a batch size of 16 achieved the highest accuracy for trajectory prediction, with a mean absolute haversine error of 187.78 on the test set. This model was used to form prediction regions, and collision prediction methods were applied to these regions to classify events as collisions or non-collisions. Sensitivity and specificity metrics revealed boundary-based methods performed poorly (20–30% sensitivity) due to a pass-by problem caused by prediction inaccuracies. Adjusting the Z-score to 3 in the Z-score bounding box intersection area method improved sensitivity to 50%, but non-boundary approaches performed better. The GMM method achieved 100% sensitivity and 94.56% specificity, while the author's modified silhouette score reached 100% sensitivity and 99.59% specificity. However, the latter's results were influenced by an empirical threshold-setting technique requiring further research. Therefore, the GMM method is recommended as the most robust approach.

4. Finally, these methods were applied to a real-case scenario where on the 13th of December, 2021 two vessels collided. The data was prepared in a manner so that the sequences with 3, 10, and 20 minutes before collision were available. As the data set with 3 minutes prior to collision had the trajectory path set to the other vessel, only ellipsoidal point intersection and Z-score = 2 bounding box area intersection methods were unable to identify the upcoming collision. However, as there was no anomalous behaviour in the cases of 10 and 20 minutes prior to the collision, boundary based as well as modified silhouette score methods were unable to identify the collision, making the GMM intersection method superior to them as it was able to correctly assess the collision risk in all cases.

Discussion and Future Work

While this thesis has addressed the collision detection problem, certain limitations remain that present opportunities for further investigation.

Firstly, enhancing the precision of trajectory prediction could significantly improve the accuracy of collision detection. Exploring alternative methods for trajectory prediction, as demonstrated in "Application of coordinate systems for vessel trajectory prediction improvement using recurrent neural networks" [8], could offer valuable insights and advancements.

Secondly, the pass-by problem could be further mitigated by employing a more frequent resampling technique, which may reduce inaccuracies arising from trajectory and speed predictions.

Thirdly, the computational performance of collision detection methods require deeper investigation. Although the Gaussian Mixture Model (GMM) demonstrated strong results, empirically its runtime was relatively high. Comparing computational efficiency across different approaches could identify optimizations and improve practical applicability.

Lastly, the novel silhouette based method should be further improved to receive a more general threshold as currently it is based on the lowest mock collision score which proved to be too high in the real-case analysis.

Code

The implementation of the methodologies and experiments conducted in this thesis has been made available as open-source code to ensure transparency and reproducibility of the research. The code can be accessed at the following Github repository: https://github.com/kasparaskaukas/master-thesis-codebase.

Acknowledgment of External Assistance Tools

In the preparation of this thesis, external tools were utilized to enhance the quality and clarity of the work. ChatGPT, an AI-based language model, was employed for editorial purposes, including refining the structure and phrasing of the text. Additionally, Grammarly was used to review the thesis for grammatical accuracy and consistency. Additionally, Kaggle was used for its GPU-accelerated environment, significantly accelerating the process of trajectory prediction modeling and testing. These tools were used solely to support the technical and editorial aspects of this thesis, while all intellectual contributions and findings remain my own.

References

[1] S. Abe, R. Thawonmas, and M. Kayama. A fuzzy classifier with ellipsoidal regions for diagnosis problems. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, 29(1):140–148, 1999.

[2] European Maritime Safety Agency. Annual overview of marine casualties and incidents 2023.

[3] I Made Asana, I Made Widyantara, Linawati, and Dewa Made Wiharta. Mahalanobis distance based dbscan for vessel near collision detection. *2023 IEEE 7th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, page 337–342, Nov 2023.

[4] Danish maritime authority, historical ais data. https://web.ais.dk/aisdata/.

[5] Da-wei Gao, Yong-sheng Zhu, Jin-fen Zhang, Yan-kang He, Ke Yan, and Bo-ran Yan. A novel mp-lstm method for ship trajectory prediction based on ais data. *Ocean Engineering*, 228:108956, May 2021.

[6] Miao Gao, Guoyou Shi, and Shuang Li. Online prediction of ship behavior with automatic identification system sensor data using bidirectional long short-term memory recurrent neural network. *Sensors*, 18(12):4211, Nov 2018.

[7] Katharine Hay. Crew on ship involved in fatal collision 'exceeded drugs and alcohol limit'. *Independent*, Dec 2021.

[8] Robertas Jurkus, Julius Venskus, and Povilas Treigys. Application of coordinate systems for vessel trajectory prediction improvement using a recurrent neural networks. *Engineering Applications of Artificial Intelligence*, 123:106448, Aug 2023.

[9] Philipp Last, Martin Hering-Bertram, and Lars Linsen. Interactive history-based vessel movement prediction. *IEEE Intelligent Systems*, 34(6):3–13, Nov 2019.

[10] Huanhuan Li, Hang Jiao, and Zaili Yang. Ais data-driven ship trajectory prediction modelling and analysis based on machine learning and deep learning methods. *Transportation Research Part E: Logistics and Transportation Review*, 175:103152, Jul 2023.

[11] Zihao Liu, Zhaolin Wu, and Zhongyi Zheng. A novel model for identifying the vessel collision risk of anchorage. *Applied Ocean Research*, 98:102130, May 2020.

[12] The European Commission, maritime transport statistics-freight transport statistics - modal split. https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Freight_transport_statistics_-_modal_split.

[13] The European Commission, maritime transport statistics-transport of goods - quarterly data. https://ec.europa.eu/eurostat/statistics-explained/index.php/Maritime_transport_of_goods_-_quarterly_data.

[14] K. Markou, L. Tsochatzidis, K. Zagoris, A. Papazoglou, X. Karagiannis, S. Symeonidis, and I. Pratikakis. A convolutional recurrent neural network for the handwritten text recognition of historical greek manuscripts. *Pattern Recognition. ICPR International Workshops and Challenges*, page 249–262, 2021.

[15] Ibomoiye Domor Mienye, Theo G. Swart, and George Obaido. Recurrent neural networks: A comprehensive review of architectures, variants, and applications. *Information*, 15(9):517, Aug 2024.

[16] Duc-Duy Nguyen, Chan Le Van, and Muhammad Intizar Ali. Vessel trajectory prediction using sequence-to-sequence models over spatial grid. *Proceedings of the 12th ACM International Conference on Distributed and Event-based Systems*, Jun 2018.

[17] Mangesh Nichat. Landmark based shortest path detection by using a* algorithm and haversine formula. 04 2013.

[18] Jane Oruh, Serestina Viriri, and Adekanmi Adegun. Long short-term memory recurrent neural network for automatic speech recognition. *IEEE Access*, 10:30069–30079, 2022.

[19] Douglas A Reynolds et al. Gaussian mixture models. *Encyclopedia of biometrics*, 741(659-663), 2009.

[20] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, Nov 1987.

[21] Alaa Sagheer and Mostafa Kotb. Unsupervised pre-training of a deep lstm-based stacked autoencoder for multivariate time series forecasting problems. *Scientific Reports*, 9(1), Dec 2019.

[22] Nikolaj Skydsgaard, Johan Ahlander, and Niklas Pollard. Crew member found dead after danish, british boats collide in baltic. *Reuters*, Dec 2021.

[23] Smita Srivastava, Lovneet Kumar, R. Jeyanthi, K. Deepa, and Vinay Aggrawal. Framework for ship trajectory forecasting based on linear stationary models using automatic identification system. *Procedia Computer Science*, 218:1463–1474, 2023.

[24] Bowen Sui, Jianqiang Zhang, and Zhong Liu. A real-time ship encounter collision risk detection approach in close-quarters situation. *Measurement and Control*, 56(9–10):1613–1625, May 2023.

[25] Yongfeng Suo, Wenke Chen, Christophe Claramunt, and Shenhua Yang. A ship trajectory prediction framework based on a recurrent neural network. *Sensors*, 20(18):5133, Sep 2020.

[26] Julius Venskus, Povilas Treigys, and Jurgita Markevičiūtė. Unsupervised marine vessel trajectory prediction using lstm network and wild bootstrapping techniques. *Nonlinear Analysis: Modelling and Control*, 26(4):718–737, Jul 2021.

[27] Chao Wang and Yuhui Fu. Ship trajectory prediction based on attention in bidirectional recurrent neural networks. *2020 5th International Conference on Information Science, Computer Technology and Transportation (ISCTT)*, Nov 2020.

[28] Chengkai Zhang, Junchi Bin, Wells Wang, Xiang Peng, Rui Wang, Richard Halldearn, and Zheng Liu. Ais data driven general vessel destination prediction: A random forest based approach. *Transportation Research Part C: Emerging Technologies*, 118:102729, Sep 2020.

[29] Xiaocai Zhang, Xiuju Fu, Zhe Xiao, Haiyan Xu, and Zheng Qin. Vessel trajectory prediction in maritime transportation: Current approaches and beyond. *IEEE Transactions on Intelligent Transportation Systems*, 23(11):19980–19998, Nov 2022.