Beyond geofencing: Behavior detection using AIS

Raphael Sturgis, Valentin Emiya, Basile Couëtoux, Pierre Garreau

Highlights

Beyond geofencing: behavior detection using AIS

Raphael Sturgis,Valentin Emiya,Basile Couetoux,Pierre Garreau

- Vessel behavior detected from AIS with machine learning models.
- Data augmentation or engineering of unbiased features allow us to generate global models.
- Comparison of different models for behavior detection such as HMM, LSTM and transformer networks.

Beyond geofencing: behavior detection using AIS

Raphael Sturgis[a,b,*], Valentin Emiya[a], Basile Couetoux[a] and Pierre Garreau[b]

[a]Aix Marseille Univ, CNRS, LIS, Marseille, France
[b]Searoutes, C/O ZEBOX, 61 Bd des Dames, Marseille, 13002, PACA, France

ARTICLE INFO

Keywords:
behavior detection
trajectory segmentation
data augmentation
AIS
deep learning

ABSTRACT

This research paper proposes two different methods for removing biases from geo-spatial trajectories obtained from AIS, enabling the creation of machine learning models that generalize well to areas with no training data. We focus here on the task of behavior detection such as moored, underway or drifting from AIS. The first method utilizes data augmentation techniques specifically designed for augmenting geo-spatial trajectories that include angled information such as direction, while the second method is based on engineering features that eliminate these geographical biases. These two methods are compared using different types of machine learning models, including random forests, hidden Markov models, LSTMs, and transformer networks.

1. Introduction

The International Maritime Organization (IMO) introduced in 2004 the automatic identification system (AIS) as a way of sharing information between vessels at sea and also with vessel traffic services. The main purpose of the system was initially as an aid to navigation and to assist vessel traffic services to avoid collisions. Since then, AIS data has attracted an increased interest from researchers and companies as treating this information can be used for other purposes such as trajectory prediction [14, 32, 9], vessel route extraction [11, 35, 13], anomaly detection [35, 33] and trajectory clustering [30, 29, 26]. Some of these tasks, such as vessel route extraction and anomaly detection, require some form of behavior detection.

Companies perform behavior detection for non fishing vessel using geofencing. This consists of identifying geographical areas were certain behaviors are expected. This can be done using polygons, and declaring an event when a vessel is within these areas. This approach can be improved by taking into account the speed. This method has two major drawbacks: firstly it is costly to create and maintain all the polygons necessary for global coverage; secondly the capabilities of these systems are limited. Indeed, it is not ideal for labelling vessel trajectories as these areas may change dynamically depending on unpredictable events such as port congestion leading to more vessels waiting at anchor to enter the port; also, some behaviors are undetectable using geofencing alone such as drifting.

These limitations with geofencing lead us to pursue systems that can perform behavior detection only using the vessel trajectory found in AIS. Such a system would enable us to perform behavior detection without the costs associated with manual labelling of geographical areas and maintenance. It could also be used to generate polygons automatically for a geofencing system.

Considering the AIS data as a time series, behaviors detection can be viewed as a task of point-wise classification over a time series. Formally, given a trajectory $T = \big((t_0, x_0), \dots, (t_n, x_n)\big)$, where $x_i \in \mathbb{R}^m$ is the feature vector gathered from AIS at timestamp t_i, the goal is to find a sequence of labels $Y = \big(y_0, \dots, y_n\big)$ describing the behavior of the vessel at each point with $y_i \in S$ the behavior of the vessel at that timestamp and S the set of possible states the vessel can be in. The aim is to find a function $F(T) \approx Y$ to estimate Y from T. We also aim to provide a model that is performant in any geographical area regardless of the availability of training data. We chose to do this in a supervised learning fashion, using a set of trajectories already labeled for training.

AIS has been used for vessel behavior detection mainly in the context of fishing vessels, in order to detect illegal fishing activities [8, 10, 20, 24, 23, 25, 22, 42, 46]. Fewer works have been done on other types of vessels [7, 28, 43]. A recent trends in literature is to use some image generation methods coupled with image recognition technics, usually Convolutional Neural Networks (CNNs) [6, 7, 28]. This has the drawback of losing some sequential aspect of the data since the time aspect is removed when generating an image. Other technics include hidden markov models (HMM) [10, 46] and recurrent Neural Networks (RNN) [24, 25], both of which model the data as a sequence. RNNs are a typical method for time series data and have been used effectively in different fields such as Natural Language Processing (NLP) [18, 21, 48] or music composition [12]. Currently the state of the art for working with times series would appear to be Transformer Networks (TN). First introduced in [45] it is commonly used for many task involving time series such as trajectory forecasting [16] or NLP [2]. To our knowledge, these models have never been experimented on for AIS behavior detection.

Most models found in the literature are trained using data from a single geographical area and tested with data from that same area [6, 22, 31]. When multiple regional data sets are used, inter-region performance is, to our knowledge, never tested [27]. This raises the problem of geographical

*Corresponding author
raphael.sturgis@lis-lab.fr (R. Sturgis); valentin.emiya@lis-lab.fr (V. Emiya); basile.couetoux@lis-lab.fr (B. Couetoux); pierre@searoutes.com (P. Garreau)
ORCID(s):

biases where models perform well in a particular area but are not able to generalize well to data from other geographical locations. This problem has been explored before in [23] for detecting human transportation modes from GPS traces collected by people carrying GPS receivers. Their approach was to remove biases using feature engineering: from the GPS traces, speed is extracted and further derived using mean and standard deviation of speed over a segment of trajectory. Furthermore, although much work has been done for fishing vessel behavior detection and some for general vessel behavior, detection little work focuses specifically on container vessel behaviors. In this paper we will focus directly on classifying these vessel behaviors. Performing behavior detection on container vessels specifically is mainly motivated by logistics, for example, to study more closely port congestion, or gain knowledge of when a vessel arrives in port automatically. Container vessel behaviors are more subject to geographical biases because of the effects of ports. Indeed, near ports we find some of the interesting behaviors exhibited by container vessels, such as anchoring and docking. Furthermore, ports come with different orientations depending on where they are located.

As stated above, feature engineering may help to remove biases. Data augmentation is another common method for removing sampling bias in data [44, 37, 41]. It is especially used when working on computer vision tasks. The main idea of data augmentation is to generate artificially unobserved data. This can be achieved in different ways, in image classification the classical ways of doing this is to flip and rotate the images.

The objective of this research is to compare different technics for removing geographical bias in order to improve vessel behavior classification. For this purpose two different methods are proposed: a novel data augmentation technic that augments the data in order to improve performance of deep learning methods, and generation of unbiased features generated from the base AIS features but without any geographical biases.

In section 2 we introduce our two strategies that can be applied on AIS data to overcome geographical biases. In section 3 we will discuss multiple decision making systems that can be used for deciding the behavior of a vessel. Finally, in section 4 some experiments comparing the different data manipulations and system combinations are proposed[1].

2. Strategies for geographical bias removal

In section 2.1 we introduce the information contained in AIS data and the concept of geographical bias. We then propose two strategies to remove geographical biases from the data. The first strategy, presented in section 2.2, is a method for data augmentation designed for geospatial sequences. The second strategy, presented in section 2.3, addresses the design of unbiased features.

[1]This paper is a continuation of [43]. The main differences are: in section 2.1 we detail the notion of geographical biases; in section 2.2 we introduce a new data augmentation technique; section 2.3 we reformulate the design of unbiased features already introduced in [43]; in section 3 we introduce new models: random forests, hidden markjov models and transformer networks; finally, in section 4, all experiments are new and are conducted using new data.

Field	unit
Timestamp	UTC
MMSI	ID
Position	degrees for latitude and longitude
Speed over ground (SOG)	knots
Course over ground (COG)	degrees from north
Heading	degrees from north

Table 1
AIS data content

2.1. Geographical biases in base AIS features

AIS is based on multiple message types each with different information and transmission rates. For this study we are mostly interested in AIS messages types 1–3 which are dynamical position reports. These messages are typically sent roughly every 2–10 seconds when the vessel is moving and every 3 minutes when the vessel is either at anchor or moored with a speed lower than 3 knots. The transmission rate of these messages is not regular and is affected by multiple external factors such as the current vessel state or the amount of vessel around using the same frequencies. The features of interest available from these dynamical position reports are summarized in table 1. More details can be found in the official specifications of the standard [40].

The timestamps and the MMSI are used to extract vessel trajectories. A trajectory is a time-series where each sample contains a set of latitude, longitude, SOG, COG and heading values. A few examples of such trajectories can be seen in figure 1.

When acquiring AIS data in a specific zone, we can observe that the local geography impacts the statistical distribution of the data. In figure 1 for example, heading angles are often aligned with the east-west orientation shown by the majority of docks, explaining the spikes at 90° and 270° in the probability distribution. One dock is oriented north-south explaining the smaller spike seen at 180°. The remaining points are concentrated around 120° and 300° which is related to the angle of the main navigation corridor oriented from north-west to south-east. This results in biases in those angular features since these specific orientations would not make sense in other areas. Latitude and longitude are also biased by the boundary of the area as well as the geographical specificities of the area (docks, anchoring zones, etc…). Only SOG is not geographically biased. These geographical biases affects training of behavior detection models and can degrade generalization performance in other areas. It is important to consider when designing models that are intended to perform well anywhere on earth, even in areas with no labeled data available. A key challenge is therefore to exploit the available information to its fullest potential without suffering from these biases.

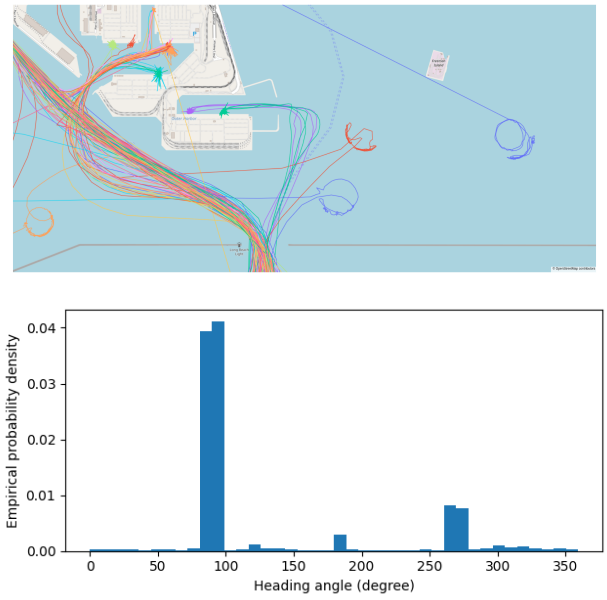


Figure 1: Top:vessel trajectories near the port of Long Beach, California, USA. One can observe typical navigation routes, concentration of points near docks and vessel waiting at anchor outside of the navigation routes (circular patterns). Bottom: empirical probability density in this area, showing two privileged orientations (biases).

2.2. Data augmentation

Data augmentation is a common technique to increase performances of machine learning models by removing biases [41]. The main idea of data augmentation is to add different views of each data sample, in a way that removes some biases from the dataset. A typical example is in image classification where random rotation, rescaling, translation and other transformations are applied to generate new training images. Those images keep the same label as theses transformations are chosen to produced images that are plausible for this class.

We introduce a new data augmentation technique suitable for AIS data. The objective is to produce a new trajectory somewhere else on the globe with modifications analogous to rotations and translations for images. The challenge is to keep coherence between the COG, heading and geographical position while moving a trajectory on the surface of a sphere.

The rest of this section will be presented as follows. Section 2.2.1 introduces the geometrical tools needed for AIS data augmentation. Section 2.2.2 shows how data augmentation can then be performed.

2.2.1. Geometrical basis

Vectors and reference frames: Let us consider a vector $\vec{v}$. It can be expressed in several reference frames $(\vec{x}, \vec{y}, \vec{z})$ and $(\vec{x'}, \vec{y'}, \vec{z'})$ as $\vec{v} = v_x\vec{x} + v_y\vec{y} + v_z\vec{z} = v'_x\vec{x'} + v'_y\vec{y'} + v'_z\vec{z'}$. We denote by $\mathbf{v} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}$ and $\mathbf{v'} = \begin{bmatrix} v'_x \\ v'_y \\ v'_z \end{bmatrix}$ the coordinate vectors of $\vec{v}$ in reference frames $(\vec{x}, \vec{y}, \vec{z})$ and $(\vec{x'}, \vec{y'}, \vec{z'})$ respectively. They are related by $\mathbf{v'} = M\mathbf{v}$ where M is the transfer matrix defined by:

$$M = \begin{bmatrix} \vec{x}\cdot\vec{x'} & \vec{y}\cdot\vec{x'} & \vec{z}\cdot\vec{x'} \\ \vec{x}\cdot\vec{y'} & \vec{y}\cdot\vec{y'} & \vec{z}\cdot\vec{y'} \\ \vec{x}\cdot\vec{z'} & \vec{y}\cdot\vec{z'} & \vec{z}\cdot\vec{z'} \end{bmatrix} \tag{1}$$

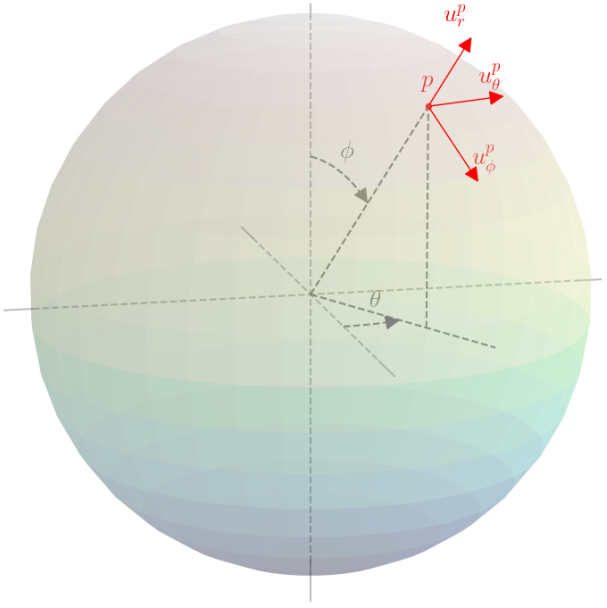


Figure 2: Representation of local reference frame vectors.

Unit vectors in spherical coordinates: For each point p on the unit sphere with spherical coordinates $(r = 1, \theta, \phi)$, there is an associated set of orthogonal vectors $(\mathbf{u}_r^p, \mathbf{u}_\theta^p, \mathbf{u}_\phi^p)$ that can be used as a reference frame, see figure 2. Their cartesian coordinates are:

$$\mathbf{u}_r^p = \begin{bmatrix} \cos(\theta)\cos(\phi) \\ \cos(\theta)\sin(\phi) \\ -\sin(\theta) \end{bmatrix} \tag{2}$$

$$\mathbf{u}_\theta^p = \begin{bmatrix} -\sin(\phi) \\ \cos(\phi) \\ 0 \end{bmatrix} \tag{3}$$

$$\mathbf{u}_\phi^p = \begin{bmatrix} \sin(\theta)\cos(\phi) \\ \sin(\theta)\sin(\phi) \\ \cos(\theta) \end{bmatrix} \tag{4}$$

Vector rotation using quaternions: This paragraph focuses on how to move points or vectors on the surface of a sphere. Without loss of generality, we will consider the unit sphere.

This transformation is obtained thanks to the rotation around an axis $\vec{w}$ with an angle α. The image of unit vector $\vec{v}$ is denoted $\vec{v'}$ This operation can be implemented with quaternion multiplication, which can perform any rotation of a

vector in dimension 3, and has the advantage of not suffering from gimbal lock unlike other vector rotation methods.

A quaternion is an hypercomplex number of the form $q = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$ with $a, b, c, d \in \mathbb{R}$ and $\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$ specials number with the following properties: $\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = -1$, $\mathbf{ij} = -\mathbf{ji} = \mathbf{k}$, $\mathbf{jk} = -\mathbf{kj} = \mathbf{i}$, $\mathbf{ki} = -\mathbf{ik} = \mathbf{j}$. An alternative scalar + vector notation in the geometrical context is:

$$q = a + \begin{bmatrix} b \\ c \\ d \end{bmatrix} \tag{5}$$

To perform a rotation of angle α around axis $\vec{w}$, we define a rotation quaternion q_r and its inverse $\bar{q}_r$ as:

$$q_r = \cos(\alpha/2) + \sin(\alpha/2)\vec{w} \tag{6}$$

$$\bar{q}_r = \cos(\alpha/2) - \sin(\alpha/2)\vec{w} \tag{7}$$

A quaternion representing $\vec{v}$ is also define using 8.

$$q_{\vec{v}} = 0 + \mathbf{v} \tag{8}$$

A rotated vector $\vec{v'}$ can be calculated by first calculating its associated quaternion and by then using the hamiltonian product as:

$$q_{\vec{v}'} = q_r q_{\vec{v}} \bar{q}_r = 0 + \mathbf{v}' \tag{9}$$

2.2.2. *Point rotation*

A feature vector x_i at time t_i of a trajectory, as described in section 1, typically contains a position p_i given in spherical coordinates $p_i = (1, \theta, \phi)$, and an angle γ_i such as heading or COG. We show how to compute a new position p'_i and a new angle γ'_i when performing a rotation with an angle α around an axis $\vec{w}$. produces a new position trajectory $T' = \{(p'_0, \gamma'_0), \dots, (p'_n, \gamma'_n)\}$. Figure 3 show a representation of this process.

The process we use for this task is the following and a schematic view is proposed on figure 3: firstly we calculate q_r and $\bar{q}_r$ for the rotation with angle α around axis $\vec{w}$ using the formulas described in paragraph 2.2.1. Using these rotation quaternions we rotate each point p_i to obtain p'_i. Then, we rotate each angles γ_i to get the angles γ'_i using the following process. First we express γ_i as a vector $\vec{\gamma}_i$ in the unit reference frame associated with point p_i as:

$$\vec{\gamma}_i = -\cos(\gamma_A)\mathbf{u}_r^p + \sin(\gamma_A)\mathbf{u}_\theta^p \tag{10}$$

Then $\vec{\gamma}_i$ is defined in the original reference frame using the matrix multiplication introduced in paragraph 2.2.1. Once defined in that reference frame it can be rotated by quaternion multiplication to obtain $\vec{\gamma}'_i$. Once $\vec{\gamma}'_i$ is expressed in the reference frame of point p'_i we can then easily recover the value of γ'_i.

The proposed algorithm is described in algorithm 1.

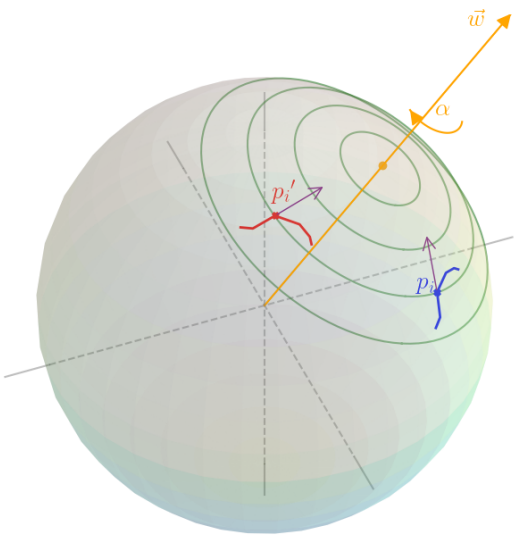


Figure 3: Rotation of a trajectory and of associated vectors. In blue the original trajectory T and in red the rotated trajectory T'. Each point is rotated along a circle (green) on the sphere centered on the axis of rotation $\vec{w}$ (orange) with an angle α. The vectors $\vec{\gamma}_i$ and $\vec{\gamma}_i$ (purple) are also represented for both points p_i and p'_i.

2.3. Engineering unbiased features

In this section we present a different approach to deal with geographical biases. For this we introduce new unbiased features that can be calculated from the base features. This was introduced in [43] and is reformulated here.

2.3.1. *Elementary operations on the features*

The biases introduced in section 2.1 are mainly due to an additive constant that affects the base features locally in time. For example the heading of a vessel is aligned with the direction of the dock while it is at port. In order to remove such biases we use subtraction operations: either through a local derivative or by subtracting a local average to measure the local spread of the feature values. We detail these operations in the case of regular features (e.g., SOG) and angular features (e.g., heading, COG, latitude, longitude).

Computing the local derivative is useful to estimate the rate of change of a given feature $f(t)$. If this feature is regular, the rate of change is usually defined as:

$$\Delta_f(t) = \frac{f(t) - f(t-\tau)}{\tau} \tag{11}$$

where τ is the sampling period. If the feature is angular, the sequence should be unwrapped before applying the same formula to avoid the discontinuities around 2π. In addition, we use the absolute value of the resulting derivative in order to ignore the difference between leftward and rightward trajectories.

The spread of a feature is given by its standard deviation. In order to calculate a local standard deviation we introduce a

Algorithm 1 Rotate geo-trajectory

Input:

- A series of points $\{p_i\}_{i=1}^n$ and angles $\{\gamma_i\}_{i=1}^n$
- A rotation axis $\vec{w}$
- A rotation angle α

Output:

- A series of rotated of points $\{p'_i = (\theta'_i, \phi'_i)\}_{i=1}^N$ and angles $\{\gamma'_i\}_{i=1}^N$

procedure ROTATE_POINT(p, q_r, $\bar{q}_r$)
  $q_p \leftarrow 0 + p$
  $0 + p' = q_r q_{p_i} \bar{q}_r$
  return p'
end procedure
procedure ROTATE_ANGLE(p, p', γ,q_r, $\bar{q}_r$)
  Let $\mathbf{u}_r^p$, $\mathbf{u}_\theta^p$, $\mathbf{u}_\phi^p$ the p-related coordinate system
  Let $\mathbf{u'}_r^p$, $\mathbf{u'}_\theta^p$, $\mathbf{u'}_\phi^p$ the p'-related coordinate system
  $M \leftarrow [\mathbf{u}_r^p, \mathbf{u}_\theta^p, \mathbf{u}_\phi^p]$
  $M' \leftarrow [\mathbf{u'}_r^p, \mathbf{u'}_\theta^p, \mathbf{u'}_\phi^p]^T$
  $\mathbf{u}^\gamma \leftarrow M \begin{bmatrix} -\cos(\gamma) \\ \sin(\gamma) \\ 0 \end{bmatrix}$
  $q_\gamma = 0 + \mathbf{u}^\gamma$
  $0 + \mathbf{u'}^\gamma \leftarrow q_r q_\gamma \bar{q}_r$
  Let γ'_i such as $\begin{bmatrix} -\cos(\gamma'_i) \\ \sin(\gamma'_i) \\ 0 \end{bmatrix} = M' \mathbf{u'}^\gamma$
  return γ'_i
end procedure
$q_r \leftarrow \cos(\alpha/2) + \sin(\alpha/2)w$
$q_r \leftarrow \cos(\alpha/2) + \sin(\alpha/2)w$
$\bar{q}_r \leftarrow \cos(\alpha/2) - \sin(\alpha/2)w$
for all $i \in \{1, \dots, N\}$ **do**
  $p'_i \leftarrow$ `rotate_point`$(p_i, q_r, \bar{q}_r)$
  $\gamma'_i \leftarrow$ `rotate_angle`$(p_i, p'_i, \gamma_i, q_r, \bar{q}_r)$
end for

sliding time window centered at the current timestamp with a radius r and compute the variance of points within this time window. For a regular feature f, the variance is classically given by:

$$\sigma_f^2(t) = \frac{1}{2r+1} \sum_{n=-r}^{r} \left(f(t + n\tau) - \bar{f}(t) \right)^2 \tag{12}$$

where $\bar{f}$ is the local mean given by:

$$\bar{f}(t) = \frac{1}{2r+1} \sum_{n=-r}^{r} f(t + n\tau) \tag{13}$$

For angular values, the definitions of mean angle and circular variance are different [39]. Given an angular feature time-series $\theta(t)$, each data point can be transposed on the unit circle as $z_\theta(t) = e^{i\theta(t)}$. One can then compute the empirical mean over a sliding window:

$$\bar{z}_\theta(t) = \frac{1}{2r+1} \sum_{n=-r}^{r} z_\theta(t + n\tau) \tag{14}$$

The circular variance is defined as:

$$\sigma_\theta^2(t) = 1 - \left| \bar{z}_\theta(t) \right| \tag{15}$$

where the mean direction is:

$$\bar{\theta}(t) = \text{angle}\left(\bar{z}_\theta(t) \right) \tag{16}$$

In equation (16), angle(z) denotes the argument of a complex number $z \in \mathbb{C}$. Note that while the samples $z_\theta(t)$ lie on the unit circle, their mean $\bar{z}_\theta(t)$ is located inside the unit circle: its modulus is close to 1 (resp. 0) when the samples are concentrated around a given direction (resp. spread over the circle).

2.3.2. Proposed unbiased features

Using the elementary operations described above, we propose a set of features that are tuned to our task. They have been selected to be unbiased and to carry useful information for behavior detection. As in [43], the following features have been selected : sog, $drift$, Δ_{sog}, Δ_{cog}, $\Delta_{heading}$, σ_{cog}, $\sigma_{heading}$, $\sigma_{\Delta_{cog}}$, $\sigma_{\Delta_{heading}}$, σ_{pos}, and σ_{sog}.

Out of these features $drift$ and σ_{pos} are not calculated using the elementary operations discussed previously. $drift$ [15] is the difference between the $heading$ of a vessel and its cog, an illustration is provided in Figure 4. σ_{pos} is the spread of the vessel position over a time window and may contribute to characterize some behaviors. It is calculated as:

$$\sigma_{pos}^2(t) = \frac{1}{2r+1} \sum_{n=-r}^{r} h^2\left(p(t + n\tau), \bar{p}(t) \right) \tag{17}$$

where the Haversine function h is a measure of the great circle distance between two positions, $p(t)$ is the position at time t given as a pair (latitude, longitude) and $\bar{p}(t)$ is the mean direction for the latitude and the longitude at time t computed using eq. (16).

3. Models for behavior detection

This section introduces existing machine learning models that are adapted for point-wise classification of time series. These models can be viewed as functions F such as $F(T) \approx Y$. they are used in section 4 to compare performance of models trained on augmented data and trained on the unbiased feature introduced in section 2.

We consider two families of models: single point classifiers in section 3.1 and point-wise classifiers for time series in section 3.2.

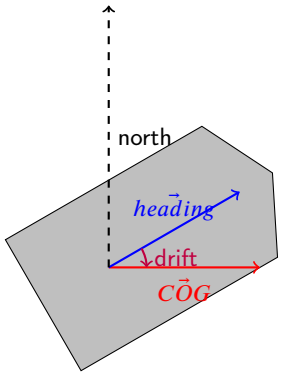


Figure 4: Graphical representation of $drift$

3.1. Single point classification

For the task of behavior classification from a time series, a strategy is to ignore the temporal aspect of the data and only focus on individual data points for classification. This allows us to use any classifier that predicts a class y from a feature vector x. In our case, at each instant, the behavior class is predicted using the feature vector x. In this paper, two well-established, simple and performant classifiers are considered in this class of models: decision trees and random forests.

Decision Tree (DT). A DT [38] is a classical supervised learning model based on a tree where each internal node is a condition on the input data x that determines which child node will be explored further and each leaf is a label y. Classification of a point x is done by descending through the tree from the root node until a leaf is reached. Given a vector describing an AIS message from our data, a hypothetical DT that might have a condition at the root discriminating on the sog with a certain threshold. If this vector has a smaller sog than the threshold then we would explore the left child of the root which might have another condition on the cog. This process is continued until we reach a leaf of the tree. Each leaf is tagged with a label which will give us the predicted behavior for this vector. DTs are fast to learn and explainable but they have some disadvantages such as poor stability.

Random forest (RF). RF [5] are a bagging ensemble method where a set of individual decision trees are trained on random samplings of the data and on a random subset of the available features. Predictions are obtained through a majority vote from the output of each tree. The resulting models are usually more robust than a simple DT with less variance on the performances and improved generalization. These models are relatively fast to learn even if slower than DTs.

3.2. Time series point-wise classification

In the previous section models that consider points of a times series individually where presented. We now focus on more specialised models that leverage the sequential nature of time series. Three different models are introduced: hidden Markov models, long short-term memory models and transformer networks.

Hidden Markov models (HMM). An HMM [3] is a statistical generative model for sequences. It relies on a Markov chain of so-called hidden states that models the non-observable internal evolution of the system. The statistical distribution of each observation is conditioned on the underlying hidden state. These models can be learned in an unsupervised fashion using the Baum-Welch algorithm [4]: it takes as an input a set of observation sequences and estimates the parameters of the Markov chain (initial and transition probabilities) and the observation probability distribution for each state. Once these parameters are fixed the Viterbi algorithm [47] allows us to predict the most likely sequence of hidden states from a sequence of observations.

In our case, we train one HMM per behavior class and merge them into a single HMM. In order to train each class-related HMM we split the labeled sequences at class boundaries. Each sub-sequence is then binned according to the unique behavior in that sub-sequence. For each behavior class i, an HMM model M_i is then trained using the Baum-Welch algorithm with the sub-sequences with class i. Once trained, we use the Viterbi algorithm to estimate the most likely state sequences for all sub-trajectories with class i.

We merge the class-related HMMs into a single HMM as follows. The new set of states is the union of the sets of states from each M_i. The state sequences obtained previously are stitched back together using the original trajectories. We infer the initial and transition probabilities from these state sequences. The observation distributions in each state remain unchanged. In each state we keep track of the label i from the class-related HMM in order to predict behavior.

Point-wise classification is obtained by estimating a sequence states from an observation sequence using the Viterbi algorithm and retrieving the associated label for each state.

Long short-term memory (LSTM). LSTMs are a family of recurrent neural network (RNN) first introduced by Hochreiter & Schmidhuber in [19]. These models can be used for sequence-to-sequence learning, sequence-to-label learning or sequences generation. Unlike basic RNNs theses models have a better ability to use information that is further in the past and have less issues with vanishing gradient through out the sequence. This makes them a good candidate for treating long sequences.

In this paper, a stack of bidirectional LSTM (bi-LSTM)[17] layers is used. The number of layers and neurons is studied in section 4.1.3. The output of the last bi-LSTM layer in the stack is then given to a 1D-CNN with a kernel size of 1 and a softmax activation to produce the prediction sequence.

Transformer network (TN). TNs are sequence-to-sequence models first introduced in [45]. These models are considered state-of-the-art for many sequence-to-sequence tasks, especially in NLP. TNs are usually composed of an encoder module that maps the input sequence to a sequential representation of same length, and a decoder module that allows

predictions of sequences of arbitrary length. In our case only the encoder module is required. TNs have been chosen here for the task of vessel behavior classification because of their state-of-the-art performances in other sequence-to-sequence tasks. They are also quicker to train than other sequence-to-sequence models such as RNNs.

A representation of the TN used here is shown figure 5. This model uses a fixed-size input sequence. The input sequence is fed through a 1D convolution in order to have an appropriate representation that is completed by a positional encoding vector. This representation is then fed to a stack of several encoder blocks.

Each encoder block, as shown in figure 5, has the standard implementation described in [45].

As LSTMs, a 1D convolution layer is finally used as the prediction layer.

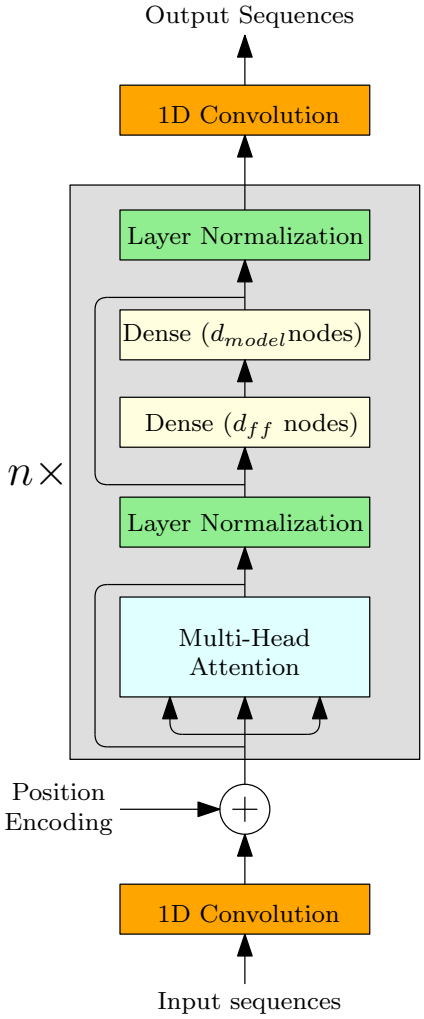


Figure 5: Transformer Network architecture

4. Experiments

In this section, experiments are conducted in order to illustrate the influence of geographical biases on behavior detection and too compare the performance of the methods we propose. All experiments and datasets are available for reproducibility[2].

4.1. Experimental setup

4.1.1. Datasets

In order to test the performances of our various approaches, two different datasets from two different geographical areas are used. This will allow us to compare the interregional performances of our models. Both these datasets were created using free and accessible marinecadastre data [34]. These datasets are created by first gathering all the data available over a set of dates and a desired geographical area and then filtering the data in order to keep only container vessels. This data is then labeled manually in the 4 classes: Underway, Moored, At Anchor and Drifting.

The first dataset contains AIS data from 98 vessels in the north-eastern part of the United States territorial waters, collected from the 1st of January 2021 until the 31st of March 2021. A visual representation of this data can be seen in figure 6a. The second dataset contains data from 96 vessels around the ports of Los Angeles and Long Beach in the south-west of the United States, from the 1st of January 2021 until the 31 of March 2021 as represented figure 6b. For simplicity these datasets are referred to as NE and LAX, respectively. The NE dataset is used for both training and testing, and therefore split into 2 subdatasets, while the LAX dataset is used only for testing. This is done in order to test the performance of models trained on the NE dataset both on data from the same area and data from a different location.

Detailed statistics for these datasets can be seen in table 2. It should be noted that all these datasets are unbalanced with varying proportions of labels. The proportions are also not constant from one region to the other.

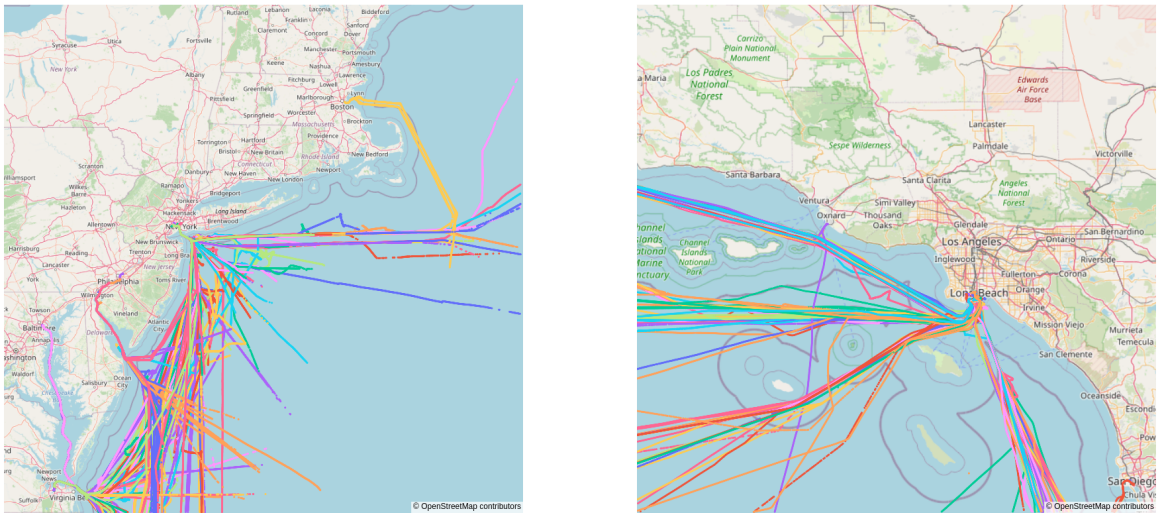
4.1.2. Data preparation

AIS data is by nature noisy. Vessels can transmit erroneous data for multiple reason such as interferences or faulty sensor for example. We clean the data during the preprocessing stage: points with out-of-specification *heading* and *cog*, and points with a *sog* more than 4 standard deviations away of the mean are discarded.

The sequence specific models require a regular sampling which is not the case with AIS. To ensure uniform sampling, irregularly-sampled AIS sequences are resampled uniformly every 60 seconds using interpolation. The continuous values are interpolated using a cubic spline interpolation while the label is interpolated to be equal to the nearest sample. Successive data points with a gap of more than 300 seconds in the original AIS sequence are split in 2 trajectories. Trajectories with less than 50 data points are discarded.

The different fields obtained from AIS are heterogeneous [23] and so are the advanced features that were described in section 2.3. For example, *sog* ranges, usually,

[2] https://gitlab.lis-lab.fr/raphael.sturgis/beyond-geofencing-behavior-detection-using-ais



(a) North-east (NE) dataset

(b) Los angeles & Long Beach (LAX) dataset

Figure 6: Overview of the trajectories in the two datasets

name	number of vessel	number of points	number of trajectories	proportion of Underway	proportion of Moored	proportion of At Anchor	proportion of Drifting
LAX	96	327901	1413	0.550	0.438	0.009	0.004
NE	98	492048	1452	0.646	0.229	0.078	0.047
NE train	80	380012	1132	0.639	0.237	0.076	0.048
NE test	18	112036	320	0.668	0.201	0.087	0.043

Table 2
Raw dataset statistics

from 0 to 30 and $heading$ from -180 to 180. It is therefore necessary to normalize these features for better performance, especially for the deep-learning models. This has been verified experimentally but is not further explored here. Each feature needs a proper normalisation method. For δ_{sog}, std_{cog}, $std_{heading}$, $std_{\delta_{cog}}$, $std_{\delta_{heading}}$, std_{pos} and std_{sog} standardization is applied; the features $drift$, δ_{cog}, $\delta_{heading}$ and $longitude$ are divided by 180 and $latitude$ by 90, i.e., by their respective theoretical maximum. Finally sog is divided by the third quartile value.

LSTMs and TNs require fixed-length sequences during the training process, which is an issue as the sequences produced so far are of arbitrary length. To work around this issue, sequences are cut with no overlap to a fixed length of 100 points and all sequences left that are shorter are padded with a masking value that is then ignored by the model during traning.

4.1.3. Training methodology

There is often a very large number of different possible hyperparameters values for each model, and they can not all be tested because of limited computing resources. Therefore, it is necessary to choose which parameters are going to be considered. The hyperparameters that were tuned for each model are summarised in table 3. To find the best parameters

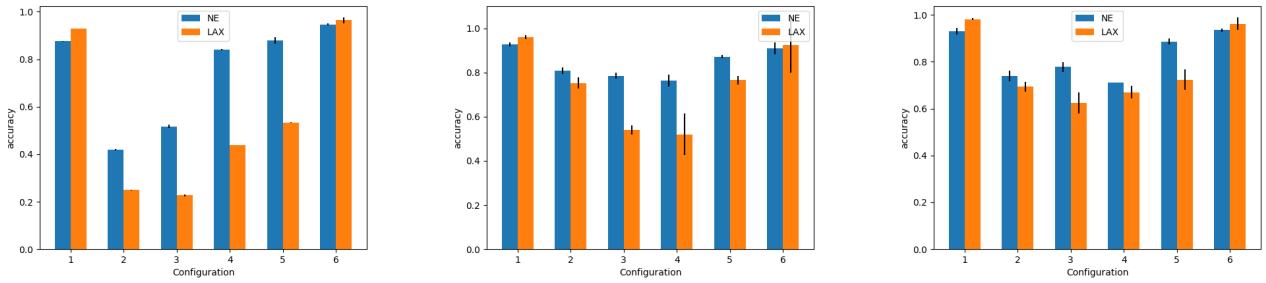
Model	Parameter	Values
DT	max depth	[1, 2, 5, 10, 20, 50]
RF	number of estimators	[5, 10, 20, 50, 100]
RF	max depth	[1, 2, 5, 10, 20, 50]
HMM	number of states for each class	[1, . . . , 20]
LSTM	number of layers	[1, . . . , 6]
LSTM	number of units per layer	[10, 20, 50, 100, 200, 500]
TN	number of layers	[1, 2, 3, 4, 5, 6]
TN	d_{model}	[32, 64, 128, 256]
TN	d_{ff}	[64, 128, 256, 512, 1024]
TN	number of heads	[4, 8, 16, 32]

Table 3
Parameters tuned for each model

a grid search is performed yielding the best combination of parameters among the values considered.

In the grid search phase, training is performed on the NE train dataset which is split into 2 datasets, a training dataset containing a random sampling of 80% of the trajectories and a validation set containing the remaining 20% of the trajectories. Each parameter combination is then tested 10 times on different splits of the dataset and the hyperparmeter combination yielding the best average accuracy on the validation set is selected as the parameters for subsequent experiments.

LSTMs and TNs are trained using the Adam optimizer on 100 epochs. Validation loss is measured at each epoch. An



(a) Accuracy using a DT (b) Accuracy using an LSTM (c) Accuracy using a Transformer network

Figure 7: Average accuracy over 10 runs of three different models with different features combinations (error bars are standard deviation of the 10 runs)

configuration	*sog*	*cog*	*heading*	(*latitude*, *longitude*)
1	✓	✗	✗	✗
2	✗	✓	✗	✗
3	✗	✗	✓	✗
4	✗	✗	✗	✓
5	✗	✓	✓	✓
6	✓	✓	✓	✓

Table 4
Feature combinations used to demonstrate the effects of geographical biases

early stopping mechanism is implemented with a patience of 10 epoch monitoring the validation loss. The learning rate starts at 0.001 and is divided by 10 every 7 epoch without improvement in validation loss. The highest batch-size possible that still allows training on the available hardware is chosen, the values selected for the LSTMs and TNs are 2 and 128 respectively.

LSTMs and TNs are implemented using tensorflow [1]. HMM are adapted from the Python library hmmlearn. Random forests and Decision trees are implemented using sklearn [36]

4.1.4. Baseline: Navigation status

AIS natively gives a navigation status field which is supposed to indicate the current vessel behavior. This field is entered manually by the vessel crew, it is therefore prone to human error and in practice it is very unreliable. Nonetheless it can be used as a baseline for our different approaches. The navigation status has 16 possible values, 4 of which are of interest to us. Status 0 is equivalent to underway in our representation, 1 is at anchor, 2 is drifting and 5 is moored.

4.2. Evidence of geographical biases

In this section, the effect of geographical biases in the base AIS features data is shown for the task of behavior detection. For this purpose, we build classifiers using different subsets of base AIS features *sog*, *cog*, *heading*, (*latitude*, *longitude*), as listed in table 4. Out of these, only *sog* is totally geographically unbiased. Therefore, we expect models trained on one geographical area with *sog* will generalize well to data in another geographical area while combinations without *sog* should suffer from poor generalization performance.

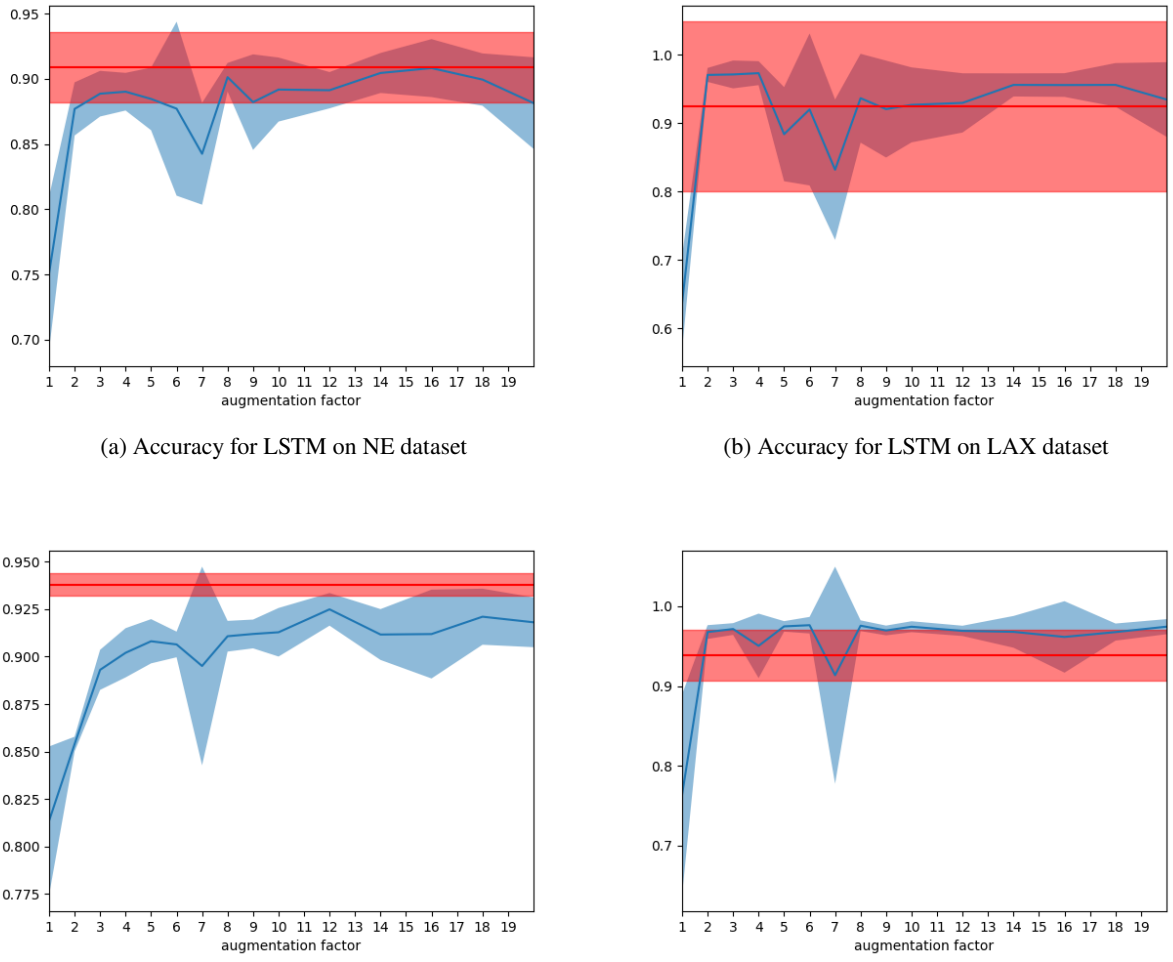
Three different models where considered: DT, LSTM, TN and the results are shown in figure 7. Every model performs best in both areas for configuration 1 or configuration 6. This suggest that *sog* is a major factor for predicting behavior. We can also observe similar performances between the two areas in those configurations, which may come from the unbiased nature of the *sog*.

For configurations 2, 3, 4 and 5 there is a significant drop in accuracy between the 2 geographical areas for each model. This suggests that indeed there exists biases in the features present in these configuration (i.e. *cog*, *heading*, *latitude*, *longitude*) and that it affects the generalization performance.

For LSTMs (figure 7b) the gap in performance is also noticeable for the configurations 3 and 4 that use *heading* and (*latitude*, *longitude*) respectively, but it not so much the case for configuration 2 (i.e. *cog*) and configuration 5 (i.e. *cog*, *heading*, *latitude*, *longitude*). This could suggests that LSTMs are less affected by the geographical biases associated with the *cog*. Nonetheless, the performance is still lower than using only *sog*. Similar conclusions can be drawn from the TNs performances (figure 7c).

We can see for all models when using *sog* as input that the accuracy is better in the LAX dataset rather than the NE dataset even thought the models were trained on the NE dataset. This suggests that the LAX dataset is simpler than the NE dataset, this can be explained by the different behavior distributions.

In conclusion, it seems that the features *cog*, *heading*, *latitude* and *longitude* are indeed biased by the local geography but that the *sog* isn't. The effects do not affect all models equally, models designed for sequential data seem to show better resilience to this phenomenon. Nonetheless, no model is able to perform significantly better using all of the features than just using the *sog*. Hence this experiment provides motivations to investigates how data augmentation and unbiased features can improve the inter-region performance.



(a) Accuracy for LSTM on NE dataset

(b) Accuracy for LSTM on LAX dataset

(c) Accuracy for TN on NE dataset

(d) Accuracy for TN on LAX dataset

Figure 8: Accuracy score vs augmentation factors (blue) and a baseline with no data augmentation (red), for TNs and LSTMs on NE and LAX datasets. The shaded area represents the standard deviation over 10 runs.

4.3. Data augmentation for reducing bias

This section presents an experiment aiming at showing the effect of data augmentation to remove geographical biases and improve behavior detection. For this, multiple LSTMs and TNs where trained on the base features, either without data augmentation or by augmenting the data multiple times. The augmented datasets are obtained by replacing the original by multiple versions of each trajectory obtained using random rotations as described in section 2.2. The number of versions for each trajectory is referred to as the augmentation factor and varies from 1 to 20.

Figure 8 shows the average accuracy score for each augmentation factor. For both models, the performance does not improve on the baseline when testing on the NE dataset. This is to be expected since in this case, the test set is in the same area as the training set: in this situation, the geographical biases contributes positively to the predictions. The more important results are found looking at performance on data from another geographical area such as LAX. In that situation, we observe that data augmentation gives no to minor improvements for the LSTM models and more significant improvements for TN models. This would indicate that the effect of geographical biases might be attenuated by this method. We also notice that the augmentation factor does not seem to have any real impact for values between 10 and 20; for values smaller than 10 there seems to be more variance in the results.

4.4. Model performances comparison

In this paper two strategies were proposed in order to remove the effects of geographical biases, a first method based on data augmentation introduced in section 2.2, a second one based on designing unbiased feature explained in section 2.3. This section seeks to compare these two strategies each one including different models, as well as models that use only base features (sog, cog, $heading$, $latitude$, $longitude$). The performance of the navigation status provided by the AIS is also used as a baseline. In order to perform this comparison we report the accuracy and macro f1 score of different models and input configurations in table 5. In the

Model	Accuracy LAX	Macro F1 LAX	Accuracy NE	Macro F1 NE
Navigation Status	0.900 (0.000)	0.747 (0.000)	0.926 (0.000)	0.733 (0.000)
Decision Tree with Base Features	0.982 (0.000)	0.494 (0.000)	0.945 (0.000)	0.861 (0.000)
Random Forest with Base Features	0.784 (0.158)	0.380 (0.099)	0.971 (0.002)	0.910 (0.006)
LSTM with Base Features	0.924 (0.124)	0.645 (0.134)	0.909 (0.027)	0.740 (0.155)
Transformer Network with Base Features	0.939 (0.032)	0.660 (0.048)	0.938 (0.006)	0.866 (0.012)
Decision Tree with Unbiased Features	0.985 (0.000)	0.810 (0.003)	0.983 (0.000)	0.964 (0.000)
Random Forest with Unbiased Features	**0.992** (0.000)	**0.857** (0.001)	**0.989** (0.000)	**0.978** (0.000)
HMM with Unbiased Features	0.977 (0.014)	0.738 (0.037)	0.966 (0.020)	0.927 (0.036)
LSTM with Unbiased Features	0.985 (0.003)	0.797 (0.025)	0.985 (0.002)	0.970 (0.003)
Transformer Network with Unbiased Features	0.988 (0.006)	0.837 (0.055)	0.982 (0.008)	0.962 (0.016)
LSTM with Augmented Data	0.927 (0.052)	0.618 (0.050)	0.892 (0.023)	0.733 (0.071)
Transformer Network with Augmented Data	0.975 (0.006)	0.679 (0.051)	0.912 (0.012)	0.776 (0.061)

Table 5
Average accuracy and macro F1 score results for different models and input features with the standard deviation over 10 runs in parenthesis.

case of models with augmented data, an augmentation factor of 10 was chosen based on the conclusions of section 4.3.

The first thing to note from table 5 is that the performances of the navigation status on both datasets is much lower than our better models. This shows that we are able to significantly improve performance with the methods presented here compared to the baseline.

A question that arises from this table is whether or not the features introduced in section 2.3 actually improve cross area performance. When looking at the models trained on our unbiased features and comparing their performances on the LAX and NE dataset, we can see that they all have similar accuracies but also the gaps between F1 scores are much lower than any other method. When looking at the models trained on the base features the gap in accuracy is not lower except for RFs where the accuracy deficit is very large. However, the F1 score is much lower on the LAX dataset than on the NE dataset. This would indicate that our unbiased features actually allows us to reduce the effects of geographical biases and therefore improve inter-area performance.

The best performing models seem to be the models trained with the geographically unbiased features. Their accuracy and f1 scores are significantly higher than any other model on both datasets. It should be noted though that even if accuracy seems constant between areas the Macro f1 score drops down significantly when going from NE to LAX. A possible explanation could be that these models conserve their performance better on some behaviors than others.

The best model seems to be the RF with unbiased features, it performs best on every metric. It also has very little standard deviation, alluding to very stable performances when retraining models.

Regarding models with augmented data, accuracy does not appear to improve compared to unbiased features. It does nonetheless show better accuracy than the same models with base non-augmented features on the LAX dataset. These models are nonetheless worse for behavior classification on the NE dataset. This may be due to overfitting when base features are used without data augmentation.

An interesting observation is also that the standard deviation is much higher for LSTMs with base features than for TN with base features. This would indicate that even though LSTMs achieve comparable results to TN with base features they achieve these performances less reliably.

In order to gain some deeper knowledge about performances for the different classes present in our datasets we gathered the precision and recall per label on the LAX dataset for a selection of models. This information is provided in table 6. It seems that the lackluster performance of the navigation status is due to the poor performances on vessels moored and drifting, for which very poor recall is observed. This suggests that the navigation status cannot be trusted to predict these behaviors. On the contrary, the highest precision is reached when predicting drifting; this is to be expected as the crew of a vessel rarely reports drifting when it is not the case.

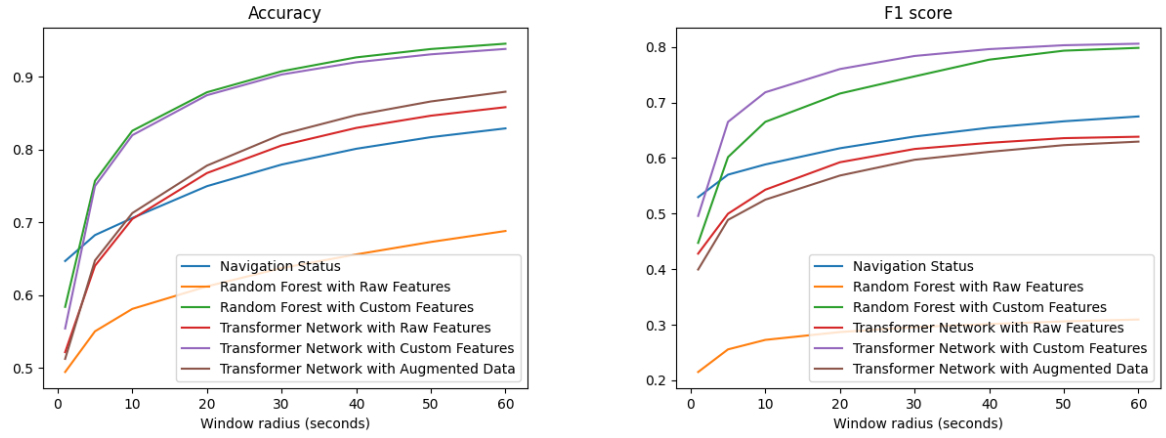
For all our models, except RF and DT with base features, precision and recall for the underway behavior are around 99% accuracy, suggesting it is the easiest behavior to reliably detect. RF using the base AIS features shows slightly less good performances for underway and moored behavior than the navigation status but is incapable to detect at anchor or drifting behaviors.

It can be noticed that the bump in performance between base and unbiased features is mostly due to better capability to detect at anchor behaviors. It also seems that RF with unbiased features are worse than any TN at detecting at anchor behaviors. This table finally shows that the best avenue for improvements is to focus on the recall for drifting events.

In the context of point wise classification, it is interesting to measure the performance near the transition between 2 classes as it should be expected that most errors would occur near these transitions. Figure 9 show the performance in terms of accuracy (figure 9a) and f1 score (figure 9b) of our

	Underway		Moored		At Anchor		Drifting	
Model	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**
Navigation Status	0.849 (0.000)	0.994 (0.000)	0.992 (0.000)	0.787 (0.000)	0.985 (0.000)	0.920 (0.000)	**0.840** (0.000)	0.141 (0.000)
Random Forest with Base Features	0.822 (0.109)	**0.998** (0.001)	0.981 (0.013)	0.538 (0.362)	0.000 (0.000)	0.065 (0.091)	0.000 (0.000)	0.000 (0.000)
Random Forest with Unbiased Features	0.992 (0.000)	0.995 (0.000)	**0.996** (0.000)	**0.995** (0.000)	0.891 (0.001)	0.908 (0.001)	0.721 (0.004)	0.430 (0.002)
Transformer Network with Base Features	0.992 (0.002)	0.990 (0.002)	0.989 (0.002)	0.883 (0.075)	0.158 (0.112)	0.695 (0.066)	0.473 (0.153)	0.540 (0.098)
Transformer Network with Unbiased Features	0.993 (0.001)	0.994 (0.001)	0.767 (0.189)	0.558 (0.106)	**0.994** (0.001)	**0.988** (0.013)	**0.840** (0.070)	**0.598** (0.070)
Transformer Network with Augmented Data	**0.996** (0.001)	0.978 (0.012)	0.301 (0.197)	0.501 (0.137)	0.986 (0.006)	0.978 (0.018)	0.372 (0.198)	0.486 (0.093)

Table 6
Per label precision and recall for a selection of models



(a) Accuracy

(b) Macro F1 score

Figure 9: Accuracy and F1 score of different models for data points within different time windows around transitions

different models for data points within different distances from transitions.

It can be noted that the Navigation Status has a flatter appearance across the different time windows than other predictors, meaning that errors are not concentrated around transitions. This is in line with our expectation as the relative poor performance of using the Navigation Status is mostly due to the fact that behaviors are completely missed by the crew and not so much that they are reported late. All other curves have a similar aspect, performances increase sharply for small time windows and then more gradually for higher values. This would suggest that these models display more errors around the transition and as we progress through the event start predicting the correct behavior. It can also be noticed that our TN and RF with unbiased features are clearly the better choice, as other models perform worse than the Navigation status just near the transition. Nonetheless, they do quickly outperform it. In terms of macro f1 score the other selected models never outperform the Navigation status.

Lastly, a focus on the performances of our models in terms of event detection is proposed. Table 7 shows the number of events found and missed by each models. Here we consider segments of trajectories were the label is constant. An event is considered detected if 90% of the individual datapoints are correctly labeled.

At anchor events seem not to be well detected as no model performs better than the Navigation Status. This is a consequence of the poor precision for this class of our models. Again, we can see that TNs seem to be better at detecting Drifting behaviors as TNs with base features and with unbiased features are on average respectively 3 times and 6 times better at detecting those events than the baseline. Nonetheless, most events stay undetected.

5. Conclusion

This study considers vessel behavior detection globally rather than in a limited geographical area. This objective is achieved by introducing the concept of geographical bias. We showed evidence of such biases in AIS data and the resulting performances drop. In order to address this issue we proposed two techniques: data augmentation and engineered unbiased features. Our experiments show that our engineered unbiased features are better for performance. Among all the models used in this study (DT, RF, HMM, LSTM, TN), the best performances are obtained with random forests and transformer networks. We detailed results for each class separately, which is an important concern when working with highly unbalanced data.

One application of this research could be the dynamical creation of semantic maps of the purposes of automatic geofencing. This could allow us to gain knowledge on active docks, generate polygons in areas with no labels and study port congestion through usage and location of anchoring areas. Another perspective of this research could be injecting additional information on top of AIS such as meteorological

Model	Underway	Moored	At Anchor	Drifting
Navigation Status	0.992 (0.000)	0.861 (0.000)	**0.889** (0.000)	0.083 (0.000)
Random Forest with Base Features	**1.000** (0.000)	0.573 (0.394)	0.022 (0.037)	0.000 (0.000)
Random Forest with Unbiased Features	0.996 (0.000)	**1.000** (0.000)	0.817 (0.025)	0.333 (0.000)
Transformer Network with Base Features	0.983 (0.004)	0.885 (0.090)	0.661 (0.072)	0.517 (0.142)
Transformer Network with Unbiased Features	0.994 (0.002)	0.997 (0.006)	0.739 (0.075)	**0.554** (0.059)
Transformer Network with Augmented Data	0.991 (0.006)	0.991 (0.011)	0.328 (0.209)	0.296 (0.106)

Table 7
Event detection statistics on LAX dataset

information or tides for improved performance. Other behaviors can also be considered, such as various fishing behaviors or refueling operations. Many interesting but rare behaviors are a challenge for most methods, this includes piracy or marine accidents. These rare behaviors could be detected using some unsupervised learning approaches. Labelling data is also very expensive even though unlabeled AIS data is plentifull. This can be overcome with unsupervised representation learning using the large amounts of unlabeled AIS data and fine tuning using the limited amount of labeled AIS data.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] K. Ahmed, N. S. Keskar, and R. Socher. Weighted transformer network for machine translation. *CoRR*, abs/1711.02132, Nov. 2017.

[3] L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.

[4] L. E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The annals of mathematical statistics*, 41(1):164–171, 1970.

[5] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[6] X. Chen, A. Kamalasudhan, and X. Zhang. An application of convolutional neural network to derive vessel movement patterns. In *2019 5th International Conference on Transportation Information and Safety (ICTIS)*, pages 939–944, 2019.

[7] X. Chen, Y. Liu, K. Achuthan, and X. Zhang. A ship movement classification based on automatic identification system (AIS) data using convolutional neural network. *Ocean Engineering*, 218:108182, 2020.

[8] B. Chuaysi and S. Kiattisin. Fishing vessels behavior identification for combating IUU fishing: Enable traceability at sea. *Wireless Personal Communications*, 115(4):2971–2993, Feb. 2020.

[9] B. R. Dalsnes, S. Hexeberg, A. L. Flåten, B.-O. H. Eriksen, and E. F. Brekke. The neighbor course distribution method with gaussian mixture models for AIS-based vessel trajectory prediction, July 2018.

[10] E. N. de Souza, K. Boerder, S. Matwin, and B. Worm. Improving fishing pattern detection from satellite AIS using data mining and machine learning. *PLOS ONE*, 11(7):e0158248, July 2016.

[11] A. Dobrkovic, M.-E. Iacob, and J. van Hillegersberg. Maritime pattern extraction and route reconstruction from incomplete AIS data. *International Journal of Data Science and Analytics*, 5(2-3):111–136, Jan. 2018.

[12] D. Eck and J. Schmidhuber. A first look at music composition using LSTM recurrent neural networks. Technical report, 2002.

[13] D. Filipiak, K. Węcel, M. Stróżyna, M. Michalak, and W. Abramowicz. Extracting maritime traffic networks from AIS data using evolutionary algorithm. *Business & Information Systems Engineering*, 62(5):435–450, 2020.

[14] M. Gao, G. Shi, and S. Li. Online prediction of ship behavior with automatic identification system sensor data using bidirectional long short-term memory recurrent neural network. *Sensors*, 18(12), 2018.

[15] M. Gao and G.-Y. Shi. Ship collision avoidance anthropomorphic decision-making for structured learning based on AIS with Seq-CGAN. *Ocean Engineering*, 217:107922, 2020.

[16] F. Giuliari, I. Hasan, M. Cristani, and F. Galasso. Transformer networks for trajectory forecasting. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 10335–10342. IEEE, Jan. 2021.

[17] A. Graves, S. Fernández, and J. Schmidhuber. Bidirectional LSTM networks for improved phoneme classification and recognition. In *Proceedings of the 15th International Conference on Artificial Neural Networks: Formal Models and Their Applications - Volume Part II*, ICANN'05, page 799–804, Berlin, Heidelberg, 2005. Springer-Verlag.

[18] A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5-6):602–610, July 2005.

[19] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, Nov. 1997.

[20] B. Hu, X. Jiang, E. Souza, R. Pelot, and S. Matwin. Identifying fishing activities from AIS data with conditional random fields. In *Annals of Computer Science and Information Systems*, pages 47–52. IEEE, Oct. 2016.

[21] Z. Huang, W. Xu, and K. Yu. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991*, Aug. 2015.

[22] K. il Kim and K. M. Lee. Convolutional neural network-based gear type identification from automatic identification system trajectory data. *Applied Sciences*, 10(11), 2020.

[23] X. Jiang, E. N. de Souza, A. Pesaranghader, B. Hu, D. L. Silver, and S. Matwin. Trajectorynet: An embedded gps trajectory representation for point-based classification using recurrent neural networks. *CoRR*, May 2017.

[24] X. Jiang, X. Liu, E. N. de Souza, B. Hu, D. L. Silver, and S. Matwin. Improving point-based AIS trajectory classification with partition-wise gated recurrent units. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 4044–4051. IEEE, May 2017.

[25] X. Jiang, D. L. Silver, B. Hu, E. N. de Souza, and S. Matwin. Fishing activity detection from AIS data using autoencoders. In *Canadian Conference on Artificial Intelligence*, pages 33–39. Springer, 2016.

[26] C. Jiashun. A new trajectory clustering algorithm based on TRACLUS. In *Proceedings of 2012 2nd International Conference on Computer Science and Network Technology*. IEEE, Dec. 2012.

[27] I. Kontopoulos, K. Chatzikokolakis, K. Tserpes, and D. Zissis. Classification of vessel activity in streaming data. In *Proceedings of the 14th ACM International Conference on Distributed and Event-based*

Systems. ACM, July 2020.

[28] I. Kontopoulos, A. Makris, and K. Tserpes. A deep learning streaming methodology for trajectory classification. *ISPRS International Journal of Geo-Information*, 10(4):250, Apr. 2021.

[29] B. Liu, E. N. de Souza, S. Matwin, and M. Sydow. Knowledge-based clustering of ship trajectories using density-based approach. In *2014 IEEE International Conference on Big Data (Big Data)*, pages 603–608. IEEE, 2014.

[30] F. Mazzarella, V. F. Arguedas, and M. Vespe. Knowledge-based vessel position prediction using historical AIS data. In *2015 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. IEEE, Oct. 2015.

[31] F. Mazzarella, M. Vespe, D. Damalas, and G. Osio. Discovering vessel activities at sea using AIS data: Mapping of fishing footprints. In *17th International conference on information fusion (FUSION)*, pages 1–7. IEEE, 2014.

[32] B. Murray and L. P. Perera. An AIS-based deep learning framework for regional ship behavior prediction. *Reliability Engineering & System Safety*, 215:107819, Nov. 2021.

[33] D. Nguyen, R. Vadaine, G. Hajduch, R. Garello, and R. Fablet. A multi-task deep learning architecture for maritime surveillance using AIS data streams. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, Oct. 2018.

[34] B. of Ocean Energy Management (BOEM), N. Oceanic, and A. A. (NOAA). Marinecadastre.gov. nationwide automatic identification system 2021. Retrieved May 04 2022 from marinecadastre.gov/data.

[35] G. Pallotta, M. Vespe, and K. Bryan. Vessel pattern knowledge discovery from AIS data: A framework for anomaly detection and route prediction. *Entropy*, 15(12):2218–2245, June 2013.

[36] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[37] L. Perez and J. Wang. The effectiveness of data augmentation in image classification using deep learning. *CoRR*, abs/1712.04621, 2017.

[38] J. R. Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[39] A. D. Rao and S. Girija. *Angular statistics*. CRC Press, 2019.

[40] M. Series. Technical characteristics for an automatic identification system using time-division multiple access in the VHF maritime mobile band. *Recommendation ITU: Geneva, Switzerland*, pages 1371–1375, 2014.

[41] C. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[42] K. L. Skaar, T. Jørgensen, B. K. H. Ulvestad, and A. Engås. Accuracy of VMS data from norwegian demersal stern trawlers for estimating trawled areas in the barents sea. *ICES Journal of Marine Science*, 68(8):1615–1620, June 2011.

[43] R. Sturgis, V. Emiya, B. Couetoux, and P. Garreau. Vessel behaviour classification from AIS without geographical biases. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 2465–2470. IEEE, Oct. 2022.

[44] D. A. van Dyk and X.-L. Meng. The art of data augmentation. *Journal of Computational and Graphical Statistics*, 10(1):1–50, 2001.

[45] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, June 2017.

[46] Y. Vermard, E. Rivot, S. Mahévas, P. Marchal, and D. Gascuel. Identifying fishing trip behaviour and estimating fishing effort from VMS data using bayesian hidden Markov models. *Ecological Modelling*, 221(15):1757–1769, July 2010.

[47] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, Apr. 1967.

[48] Y. Wang, M. Huang, X. Zhu, and L. Zhao. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 606–615, 2016.

CRediT authorship contribution statement

Raphael Sturgis: Conceptualization, Methodology, Software, Writing - original draft & editing. **Valentin Emiya:** Conceptualization, Methodology, Validation, Writing - review & editing, Supervision. **Basile Couetoux:** Conceptualization, Writing - review, Supervision. **Pierre Garreau:** Conceptualization, Writing - review, Supervision.

Acknowledgments

The research in this article was supported by Region SUD.