Maritime Anomaly Detection - A Probabilistic Deep Learning approach using GeoTrackNet and Transformers

Masters Thesis

AVS10 - Alexander F. Nielsen

AALBORG UNIVERSITY
STUDENT REPORT

Aalborg University
The Technical Faculty for IT and Design
Department of Electronic Systems

AALBORG UNIVERSITY
STUDENT REPORT

The Technical Faculty for IT and Design
Niels Jernes Vej 10, 9220 Aalborg Øst
http://www.aau.dk

Title:
Maritime Anomaly Detection - A Probabilistic Deep Learning approach using GeoTrackNet and Transformers

Project Period:
01/02-2025 to 04/06-2025

Participant(s):
Alexander Førgaard Nielsen

Supervisor:
Thomas B. Moeslund

Copies: 1

Page Numbers: 33

Date of Completion:
June 4, 2025

Abstract:

This report contains an empirical evaluation of the unsupervised Deep learning framework GeoTrackNet, originally developed for maritime anomaly detection of vessel trajectories. In contrast to the original paper, which lacked a labeled dataset, this project introduces a labeled evaluation to quantitatively asses performance of the model based on ground truths. To provide a comparison and potential improvement to this state-of-the-art model, a transformer encoder block is introduced in order to allow long-term dependency modeling using attention, inspired by models such as the TrAISformer. When both models have been trained on normal behavior and learned to recognize what normal behavior is, then a contrario detection is applied. This method uses a threshold ε to determine allowed number of false alarms. The models have been evaluated under a series of these values where classification metrics have been used to determine performance. The findings of this project indicate that performance difference between the original and extended model is negligible under curent training conditions. However, limited time for hyper parameter tuning, especially important for transformer-based models, may have hindered the potential and thus the results gathered.

Preface

During the project, generative AI tools have been utilized. The specific platforms used are Google Gemini 2.5 Flash, OpenAI GPT4o, and Claude Sonnet 4. These platforms were used to improve clarity in text, correct grammar, support brainstorming during the design phase, and to understand legacy frameworks like Tensorflow 1.x. These tools where especially useful due to the shift in framework execution and the lack of documentation for it.

Aalborg University, June 4, 2025

Contents

1 Introduction

The maritime domain generates a lot of data daily through the Automatic Identification System (AIS), which is a system that transmits critical navigational information such as position, speed, heading, and identity of a vessel [1]. AIS data plays a crucial role in enabling maritime safety both through the ability to monitor shipping lanes and movement to avoid collisions as well as intelligence gathering for avoiding sabotage or other illegal actions that might occur. While AIS plays a key role in these areas, the sheer scale, general heterogeneity, and vulnerability to noise or manual manipulation present a significant challenge in performing proper analysis.

The dynamic nature of vessel movement patterns further complicates efforts to clearly define normal and abnormal behavior. These characteristics have led to increased interest in data-driven approaches that is able to learn patterns directly from data without relying on predefined rules or thresholds. For this reason the project will investigate deep learning methods to perform anomaly detection in AIS data, focusing on the identification of trajectories deviating from normal behavior. These abnormal trajectories could indicate suspicious or otherwise unsafe behavior that could result in various kinds of accidents. Analyzing AIS data can be a challenge due to it being both spatiotemporal and high-dimensional, comprising of positional updates along with other contextual metadata such as vessel type, operational status, and destination. The reporting frequency is irregular thus resulting in frequency of received data between vessels being different. This leads to challenges in aligning and thus requires pre-processing in order to perform meaningful analysis between vessels. Furthermore, vessel behavior is highly context dependent as what might appear as anomalous for a fishing vessel might be normal for a cargo ship. All these variables makes the task of defining ”normal” behavior difficult.

These characteristics can make the popular supervised learning framework infeasible as anomalies are rare and labeled data even more so. Instead this problem calls for another approach, with unsupervised learning addressing a lot of the issues present with labels in AIS data. Unsupervised learning aims to learn from large quantities of unlabeled data in order to detect deviations from learned patterns. This project aims to explore unsupervised models and approaches to detect anomalies within AIS data packages. The main is was on the sequential nature and probabilistic representations due to the irregular and complex nature of the data. The initial goal was to propose a novel method for modeling the underlying behavior within maritime vessel behavior and thus enabling early detection of anomalies that otherwise could have been illegal activity or near collision events. Before deciding on this however a thorough study of existing solutions is needed.

While researching this area and encountering such methods and frameworks as GeoTrackNet, which utilizes recurrent neural networks (RNNs) and variational inference to model trajectories, where reviewed and tested. The existing results from methods like this one sparked an interest in shifting the project towards and empirical study of architectural extension, focusing on integrating more long-term information using attention mechanisms. Attention has been shown to improve learning of both short- and long-term dependencies in sequential data. Even with text processing being the strongest foundation for attention, it can be a promising extension to AIS trajectory models. This project will therefore focus on a comparative evaluation of the original GeoTrackNet model and an attention-extended version using a labeled AIS dataset to establish a classification benchmark during testing.

2 Problem analysis

The problem analysis will highlight context, challenges and aspects of the problem at hand that will shape a specific problem statement. This statement will be the result of technical aspects discovered as well as related works and their solutions to problems.

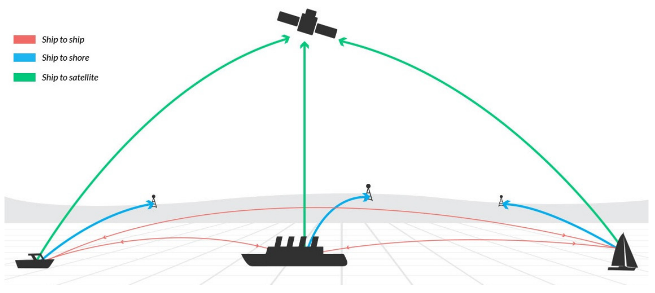
The problem analysis will firstly explore and explain what the Automatic identification system is (AIS). This section will answer question such as what AIS is, what its used for, and a surface level explanation of rules that has been defined. Afterwards a more data-centric approach is used to explore AIS as data. This section will explore the structure of AIS data revealing what information is present within, and what challenges AIS data might bring. Here a small section exploring AIS anomalies will be presented to get a deeper understanding of what an anomaly is within AIS data. Lastly some related works will be explored in order to guide the design choice of a solution and learn from challenges previous projects might have faced. This section will play a large role in guiding decisions throughout the project and will be the foundation for data-processing and outputs.

2.1 What is the Automatic Identification System

AIS is a system originally developed as a collision avoidance tool and has been the most significant development in maritime navigational safety since the introduction of radars. AIS has since been used for a variety of other purposes such as general information gathering, target tracking, and rescue operations. Expanding the system with stations placed on the shore allows port authorities and other maritime safety bodies to regulate maritime traffic. This helps to reduce risks of accidents and overall increase seafaring safety [1][2].

The general structure and communication system is illustrated in figure 2.1 and shows how communication happens between vessels, satellites, and shore stations. AIS is mandatory for certain vessels defined by the Safety of life at sea (SOLAS) treaty and adapted by the Danish Maritime Authority (DMA). SOLAS is a treaty defining the minimum standards for construction, equipment, and faring of vessels. In regards to AIS a classification is performed defining wether a ship is Class A or Class B. A list of requirements defining which vessels fit into which class is defined by chapter V, regulation 19 which states the following requirements for ships.

- Ships of more than 300 gross tonnage (GS) engaged on international voyages
- Cargo ships of more than 500 GS not engaged on international voyages
- All passenger ships irrespective of vessel size.



H

Figure 2.1: An illustration of AIS communication ways and how communication happens not only between vessels but also satelites and shore stations. Figure is from NATO-shipping center [1]

If any of the above requirements are met the vessel is categorized as a class A vessel and thus is obligated to carry an enabled AIS transceiver at all times, with only a few very specific exceptions [1][3].

With this AIS transceiver installed, vessels are now able to transmit messages to each other, to stations on the shore, and to satellites in orbit with direct visibility. This net of communication allows ships to effectively communicate positions to each other giving the helmsman more information about nearby maritime traffic and thus allows planning based on more information. This also allows external observers to gather this data and process for more general information gathering.

2.2 AIS as Data

AIS generates a lot of maritime navigational data, making it a rich resource for information regarding traffic monitoring, route planning, and behavioral analysis. Each AIS package contains structured information that is composed of multiple fields that can be categorized into three main categories. These categories are static, dynamic, and voyage specific data.

The static data will contain fields such as the ship ID, ship name, ship type, etc. which does not change over the course of a voyage or even between voyages. Dynamic fields will contain information that changes between each AIS data package and contains fields such as latitude, longitude, speed, heading, etc. And at last the voyage specific fields will contain information that is defined when the voyage starts such as destination, cargo type, estimated time of arrival, etc.

From a data-centric perspective, the dynamic fields in AIS data packages can be viewed as a form of trajectory data, where each vessel broadcasts a sequence of geolocations. Common fields found in AIS packages include but are not limited to:

- **MMSI (Maritime Mobile Service Identity)** – a unique identifier for the vessel.
- **Timestamp** – the UTC time of message transmission.
- **Latitude / Longitude** – the vessel's position.
- **Speed Over Ground (SOG)** – current velocity .
- **Course Over Ground (COG)** - current heading
- **Navigational Status** – indicator of operational mode (e.g., underway, at anchor).
- **Rate of Turn**, **Draught**, **IMO Number**, **Ship Type**, etc.

A comprehensive list of fields present in a full AIS data package obtained from the Danish Maritime Authority [4] is shown in Appendix A.

Despite its structured nature, AIS data presents a few challenges that can make it difficult to process and analyze. A few examples of these challenges is

- Irregular sampling
- missing data
- noise and outliers
- Lack of ground truths
- Heterogeneity across vessel types

As a result of these challenges, pre-pocessing steps are important to consider such as filtering and interpolation. These methods are often important as they allow for more homogeneous data structures that is more ideal for statistical analysis and machine learning pipelines. These challenges highlight the need for robust models capable of handling both uncertainty and sequences of data with variable length. This motivates the exploration of deep learning models that show a promising direction for AIS based maritime navigational anomaly detection.

2.3 AIS data anomalies explored

Before moving to the exploration of models and methods for anomaly detection a short analysis of these anomalies is important, to get a basic understanding and idea of what an anomaly might entail. An example of anomalous AIS data can be shown by sorting data into full trajectories with methods explained in chapter 3. In figure 2.2 a set of 25 normal voyages can be seen which generally follow fairly well defined routes and tendencies.

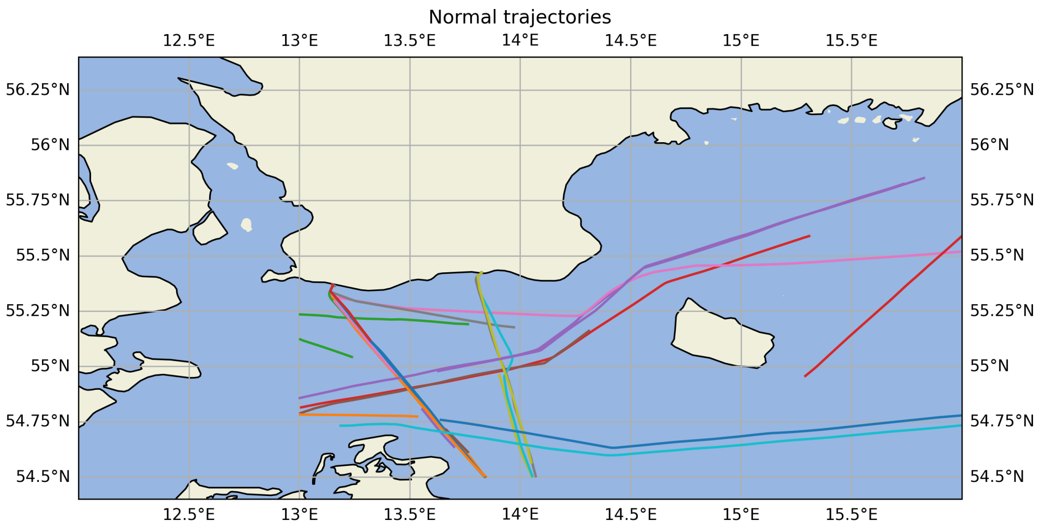


Figure 2.2: A map showing 25 normal trajectories in the region between Sjælland and Bornholm. The colored trajectories is to differentiate trajectories and does not carry any meaning

Showing 25 abnormal voyages however shows much more chaotic behavior which is much more difficult to predict. In figure 2.3 a series of 25 anomalous voyages are shown as a result of a collision event near the most densely packed area of the figure.

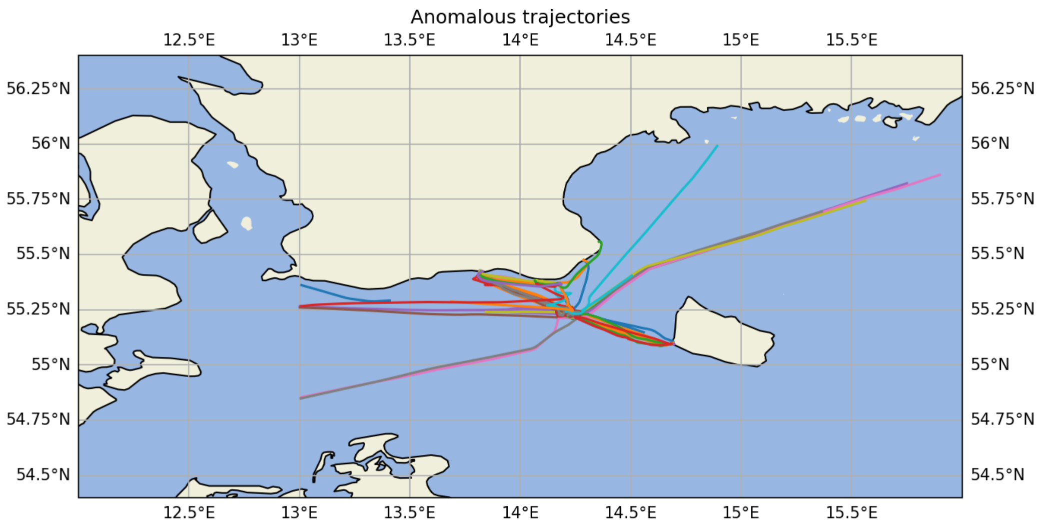


Figure 2.3: A map showing 25 anomalous trajectories in the region between Sjælland and Bornholm. The colored trajectories is to differentiate trajectories and does not carry any meaning

Analyzing other parameters such as speed can also reveal important information with normal ship trajectories having a general tendency for consistent speeds with high predictability. Anomalous trajectories however tend to have chaotic and hard to predict speeds which can change wildly as shown in figure 2.4

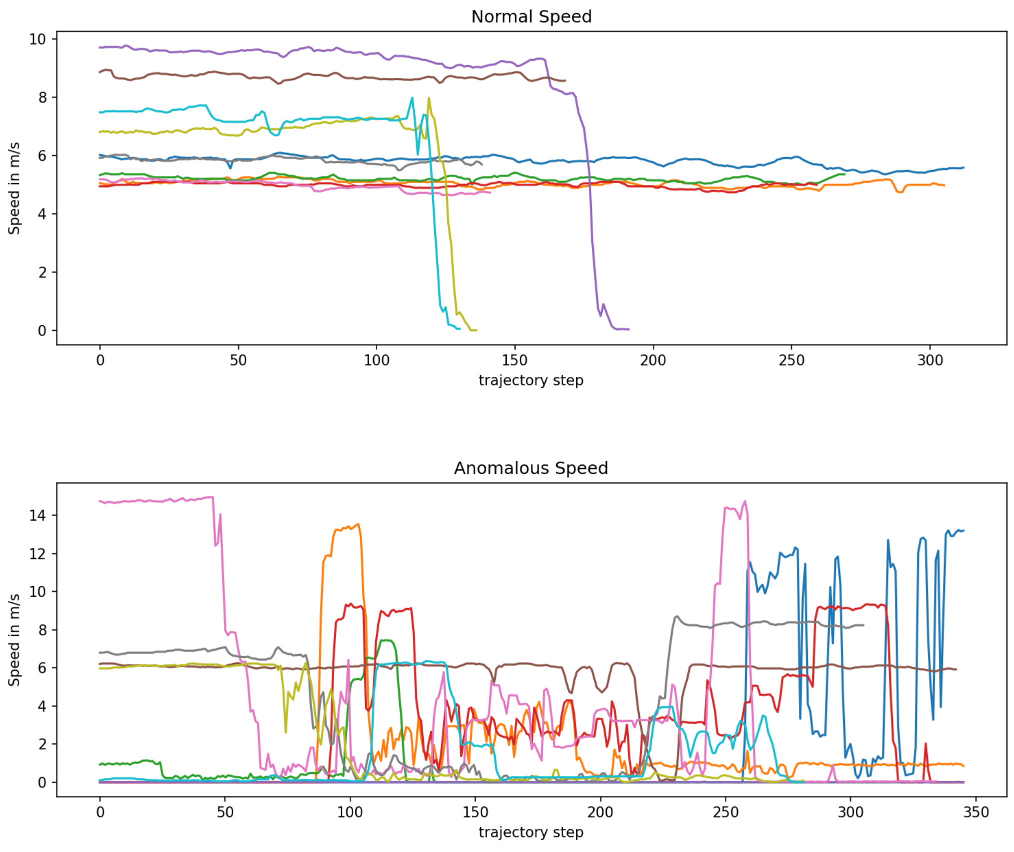


Figure 2.4: A visual comparison between normal and Anomalous speeds in 20 total trajectories, 10 normal and 10 anomalous

2.4 Related Works

Automated detection of anomalies within AIS data has been subject of extensive research. A wide range of approaches have been utilized ranging from traditional rule-based heuristics to modern deep learning algorithms utilizing a more nuanced data-driven approach. This section will present an overview of representative techniques grouped into classical machine learning and modern deep learning methods, with a focus om models and approaches relevant to this project. Before looking at specific methods an important addition to this field needs to be addressed. Within the field of AIS data processing, a general lack of labeled data is posing a major challenge to the development of anomaly detection methods as no real benchmark can be done. For this purpose the Technical University of Denmark (DTU) has in collaboration with Terma A/S created a dataset containing 521 vessel trajectories from which 25 are marked as anomalous. This dataset provides labeled data and thus opens the possibility for evaluating models on known anomalies and allowing evaluation of test performance.

2.4.1 Classical methods

Two popular methods for performing unsupervised anomaly detection using AIS data are kernel density estimation (KDE) and gaussian mixture models (GMM). These methods have warranted multiple comparison studies looking into their performances relative to each other but generally concluding that both methods produce a high amounts of both false positives and negatives [5][6]. Their reliance on relatively simple assumptions makes it difficult to capture the complex nature of maritime behavior.

Another commonly explored method is using Density-based spatial clustering of applications with noise (DBSCAN) adapted by Goodarzi et al. in 2019 [7]. They propose a two stage method consisting of an extraction of normal patterns using density based clustering algorithms. The second stage aims to detect anomalous behavior based on two separate distance measure. However, DBSCAN based approaches contain several challenges using data such as that from AIS. A key issue is the non-uniform density of received AIS data which tends to vary based on speed of the vessel [8].

2.4.2 Deep learning methods

To adress some of the limitations present in traditional approaches, deep learning methods have gained a lot of traction. Particularly With access to vast amounts of AIS data provided by the Danish Maritime Authority. Deep learning methods have gained popularity as they are able to capture more complex relations within the input data. While these methods typically require large datasets, this is well supported by the vast amount of historical data available.

One such method is the **GeoTrackNet** proposed by Nguyen et al. in 2019 [9]. This approach combines a few techniques to model the heterogynous nature of AIS data. The framework that was proposed consists of two main components. These components are a deep learning model that estimate likelihoods of trajectories, and an accompanying a contrario detection method that classifies the trajectory as anomalous or not based on the estimated likelihoods.

The process of the GeoTrackNet can be separated into three stages being pre-processing, modeling, and detection.

The input used is the latitude, longitude, speed, and course. These continuous values are discretized into a 4-hot encoded representation. This means each of the four inputs are 1-hot encoded separately and afterwards concatenated creating one feature vector [9].

The modeling step utilizes as variational recurrent neural network (VRNN) proposed by Chung et al. in 2016 [10] aiming to improve the modeling capabilities of deterministic recurrent neural networks by introducing elements originally used in variational autoencoders. This addition introduces a stochastic latent space consisting of a mean and variance instead of a deterministic vector representing a specific data point. This approach enables the model to capture both temporal dependencies and consider uncertainty in AIS data.

The detection step utilizes a special version of a contrario detection that has been separated into grid cells which aim to address issues with inconsistent confidence in the model. Some areas like open sea is expected to be more regular in behavior whereas shorelines or ports is expected to be inherently more chaotic and hard to model. This approach is referred to as geospatial a contrario detection and utilizes the models output as understanding what normal behavior is like and then evaluating the next step based on likelihood under the models prediction.

After the proposal of GeoTrackNet another way to model AIS data, not focused on anomaly detection, but rather trajectory prediction utilized the transformer architecture proposed by Nguyen et al. in 2021 [11]. The use of attention in what has been called the transformer architecture was popularised by Vaswani et al. in 2017 with the paper ”Attention is all you need”[12]. Using this transformer architecture a more robust approach to sequential data can be used to model longer term dependencies and extend the time window in which accurate predictions can be made.

The transformer architecture however is inherently deterministic and thus lacks the expressiveness of the variational latent space that the GeoTrackNet architecture relied on. This also introduces problems regarding the a contrario detection that determines wether or not a sample is considered anomalous. For this other research have been done, trying to increase the expresiveness of transformers through variational transformer architectures. One such proposed method was introduced by Lin et al. in 2020 [13] which introduces two methods of implementing a variational latent space into the transformer in order to diversify outputs. This work however is focused on natural language processing (nlp) as the transformer architecture originally was developed for translation.

An entirely different approach utilizing the generative capabilities of Generative Adversarial Networks (GANs) are Liang et al. in 2024 [14]. This method utilizes only the geographical locations of trajectories, creating a binary matrix or image representing the trajectory which the vessel is following. Their work however will not be considered throughout this project however due to a few reasons. Firstly their initial reliance on seemingly arbitrary alterations to the input data in order to simulate anomalies, is unlikely to emulated real-world examples of anomalies and thus does not support any results. This point is furthered by a skepticism for GAN-based approaches as they learn to model visually plausible samples which might not mimic real-world mechanics which dictate plausible anomalies. It is important however to acknowledge the strong results they then get from a labeled dataset which suggest a strong model able to classify anomalies with an f1-score as high as 0.8847. As this approach has been evaluated on labeled data already the focus will be on probabilistic modeling instead which will allow for more control and interpretability of outputs.

2.5 Problem Statement

The problem statement will guide the design and solution decisions made to solve the problem at hand and will be the primary factor driving the way in which the solution is evaluated. The problem statement will is based on factors such as the type of data available, the context in which it is used, and previous works that have provided valuable insight into which directions will be most effective.

With methods such as GeoTrackNet and TrAISformer explored, some immediate questions arise. GeotrackNet seems effective in its purpose but without any labeled data there is no real way of knowing its performance. using the labeled data from DTU now allows models to be compared and more careful tests can be performed. With this background the problem statement will be defined as such :

How well does a state-of-the-art method like GeoTrackNet perform when compared to actual labeled data? and would additions like attention mechanisms that make the foundation of the TrAISformer improve the performance of GeoTrackNet by enabling longer-term dependencies to be learned?

3 System design and implementation

This chapter will first introduce the details of a baseline model which in this project is the GeoTrackNet framework with a VRNN that aims to model normal[1] behavior of maritime vessels where statistical analysis is then used to decide if behavior is consistent with normal behavior or deemed unlikely and thus an anomaly.

Throughout the project, several methods of expansion was considered with only a single one having been implemented largely due to practical constraints. Specifically, the original VRNN architecture of GeoTrackNet was extended with a transformer encoder that further processes the VRNN outputs to allow for long-term dependency modeling. Further experimentation was limited by time and significant computational challenges being largely memory related. Although swap memory could theoretically be used to alleviate these challenges, doing so would result in prohibitively long training times. The combination of slower data transfer from farther away caches and CPU-based training would extend the training times, making processing of results infeasible within the time scope of this project.

3.1 General data pipeline

In any pipeline of data revolving around machine learning or deep learning, the data structure needs to be defined. The datasets used in this project is provided by ”Danmarks Tekniske Universitet” (DTU) meaning the the technical university of Denmark [15]. Two datasets are used as one is for training and contains more than a 100.000 vessel trajectories over a four month period from June through November 2021. And a dataset for testing containing 521 labeled trajectories containing 25 trajectories labeled as anomalous. This data will dictate which features and trajectories are used as locating the same 100.000 trajectories and appending extra features is infeasible in the allocated time period given the available hardware, as raw AIS data is storage intensive with two months worth of data is able to use up 100GB of storage space. An example being December 2019 and December 2020 together only containing 10.000 useful trajectories.

Even with a dataset provided, the values and content is still important to understand. In order to understand the data a thorough exploration of the data is needed. The data starts in an unsorted list of AIS messages with values in each entry being comma separated, and thus each AIS message would be a row in a ”comma separated value” (CSV) file. With the data unorganized like this, some pre-processing needs to be done. Firstly the input is processed such that individual AIS messages are sorted together and grouped based on MMSI. When these groups are then sorted by timestamp the rows can essentially be seen as a trajectory. It is important to realize however that this seemingly singular trajectory might not necessarily be considered a single trajectory. Analyzing the timestamps reveals that some time steps are fairly close matching the transponders broadcast frequency while other might be multiple hours or days between. This inconsistent behavior can be explained in many ways, with a likely explanation being an ended voyage and thus ended trajectory or other behavior that can be considered an ended voyage. An example of this can be seen table 3.1 which shows a voyage that likely ended or exited the region of interest (ROI) at around midnight, and then shows up around 16 in the afternoon.

[1] Normal will be used often throughout this chapter and should not be confused with a gaussian distribution

Date	Class	MMSI	...
28/05/2025 00:00:00,	Class A,	12345,	...
28/05/2025 00:00:05,	Class A,	12345,	...
28/05/2025 00:00:08,	Class A,	12345,	...
28/05/2025 16:19:47,	Class A,	12345,	...
28/05/2025 16:19:52,	Class A,	12345,	...
....	...	...	...

Table 3.1: Table of fictional AIS messages, aimed to show similar AIS messages that could be considered two different voyages.

Before explaining anything further it is valuable to consider these rows as samples, and consider collections of rows as trajectories.

```
sample =  [Timestamp,   type,   MMSI,   ....]
```

Grouping multiple rows and thus samples together into a single collection would then make a trajectory:

```
trajectory = [[Timestamp_1,     type_1,     MMSI_1,     ....],
              [Timestamp_2,     type_1,     MMSI_1,     ....],
              [Timestamp_3,     type_1,     MMSI_1,     ....],
              ...               ...         ...         ...
              [Timestamp_n,     type_1,     MMSI_1,     ....]]
```

Looking at the rows shown in table 3.1 a large difference is present between row 3 and 4. In this example the first 3 samples would be considered a trajectory and sample 4 and 5 would be considered another trajectory, resulting in 2 different trajectories. This means that a single MMSI can be a part of multiple trajectories throughout the dataset, which is the case for the dataset used during this project. Specifically trajectories with a difference between time steps more than 10 minutes are split in two, while discarding ones below 10 minutes. The last step for defining trajectories is splitting trajectories lasting more than 12 hours into equal segments shorter than 12 hours. This allows for more manageable sequences and shortens long trajectories to be more in line with other trajectories in the dataset.

To create a dataset, the dynamic fields are the foundation of what will be used for analysis. A few static fields is however important to save, as MMSI will allow reference back to the ship and timestamp will reference the time at which this message was sent. The dynamic fields that is of interest are the latitude, longitude, speed over ground (SOG), and course over ground (COG). The dataset used contains these values but valid arguments can be made for including fields such as heading and rate of turn (ROT) and maybe even type of cargo as that might introduce subtle changes in navigational patterns. Due to limitations of the available dataset those extra values will not be considered in this project.

Considering a list of gathered trajectories, the interval with which each sample has been received is irregular and thus can be difficult to process in any meaningful way. For this reason each track is interpolated between samples to match a specific frequency. This way it is easier and more consistent to work with the data, as it is evenly spaced in time. In order to address this a resampling and interpolation step is introduced. The specific way in which resampling and interpolation is done has been deemed confidential and thus the specifics is not known. However it is known that the data is resampled and then linearly interpolated to match a sample every 120 seconds. A general way to do it will be to define the timestamps of the first and last sample in a track, then define a point every 120 seconds between these two timestamps. The value which should be assigned to each of these points is then based on a linear interpolation between the two nearest points. An illustration of this process can be seen in figure 3.1

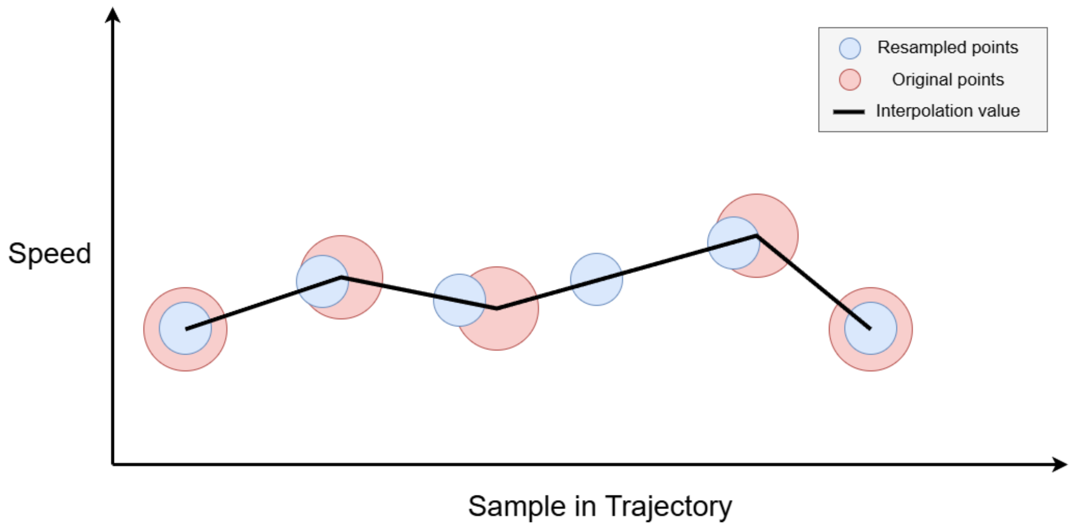


Figure 3.1: An illustration of resampling using linear interpolation to assign values to new points with speed used as an example. The evenly spaced Blue points represent the newly created samples using values which have been extracted from the linear interpolation, represented by the black line between the original points marked in red.

Now each track contains evenly spaced samples and thus more ideal for modelling and analyzing. With trajectories resampled and sorted, the final pre-processing step before inputting data to a model can be done. The final step is converting input to a categorical input for easier interpretation by the model. According to Nguyen et. al [9][11] a more ideal input structure as opposed to four real valued inputs is a categorical input constructed of four 1-Hot encoded inputs concatenated together to form a 4-Hot encoded input. The idea is illustrated in figure 3.2

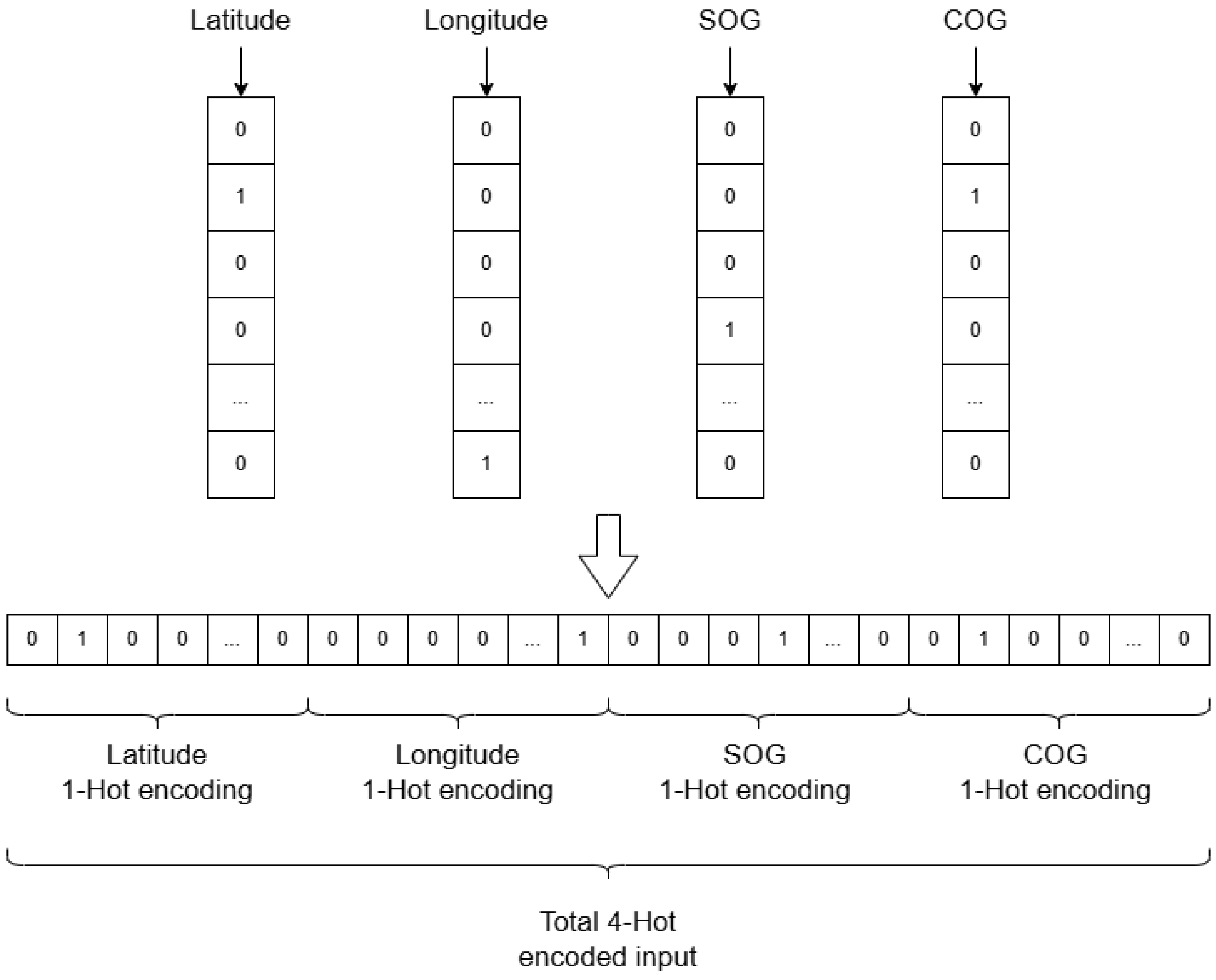


Figure 3.2: How the raw values latitude, longitude, SOG, and COG is transformed from singular values, to 1-Hot encoded vectors and at last concatenated to a total 4-Hot encoded input

This representation of input is important to understand and remember as it will be relevant once again when analyzing the output statistics during evaluation of the results.

3.2 Modelling ”Normal” Behavior

With data encoded in a more optimal way it is now possible to pass it through a machine learning model, in this case a deep learning model. GeoTrackNet is a framework which in its entirety contains both the data encoding, the model for predicting a normalcy score, and the final detection of wether a datapoint is marked as anomalous or not. This section will focus on the modeling of normal behavior and prediction of a normalcy score.

The approach used is based on a variational recurrent neural network (VRNN) which is similar to a normal RNN but with an added variational latent space. This simply means that instead of deriving a deterministic meaning in a single vector, the parameters for a distribution is estimated instead. For a normal distribution this would be the mean and variance resulting in two latent vectors. The reason for introducing a variational version of the model is due to the complex nature and external factors that might influence the behaviors which is not captured in the inputs directly. A probabilistic approach then aims to counteract this uncertainty by modeling variations in the input and thus difficult data is inherently marked with a high variance meaning a low certainty. A comparison between a regular RNN and its variational variant can be seen in figure 3.3

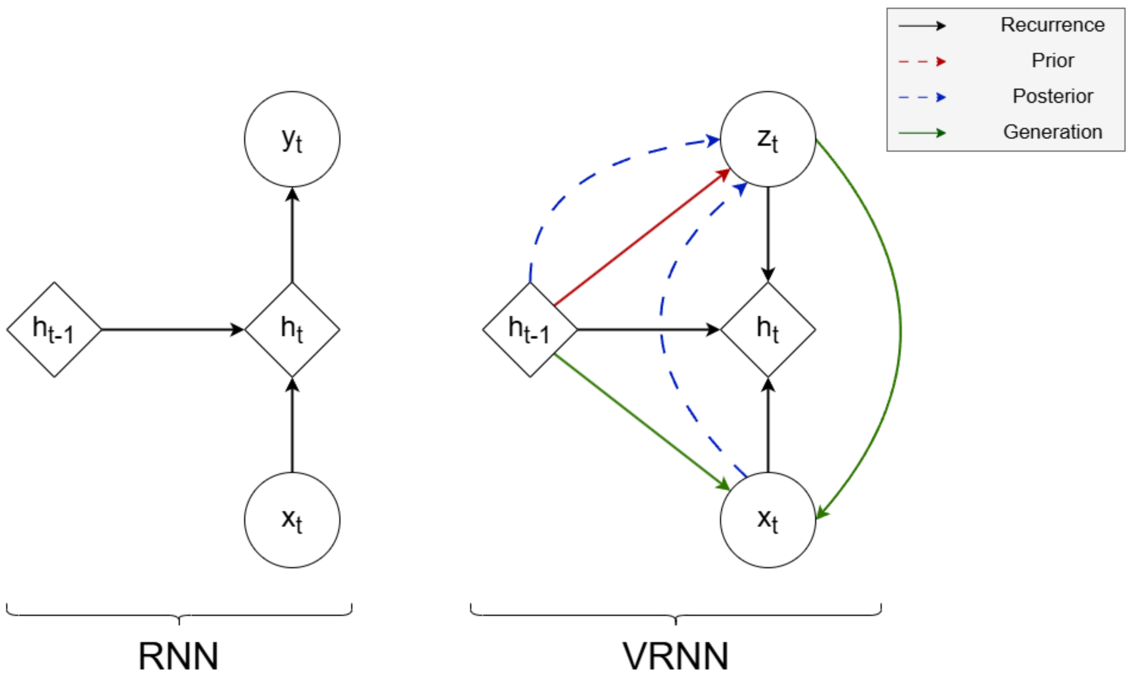


Figure 3.3: On the left a traditional RNN is shown, using the previous hidden state together with an input to generate a new hidden state and an output. On the right a VRNN is shown using the previous hidden state and the modeled latent space to generate a new sample x_t

The authors and creators of GeoTrackNet however have introduced a few extra steps within the model to achieve more robust feature extraction as recommended by the VRNN authors. Specifically an input feature extractor and a latent space feature extractor. These two multilayered perceptrons (MLPs) are used in order to allow for nuanced interpretation of input and latent space allowing for some transformation to be learned during the training process. The integration of these into the architecture can be seen in figure 3.4

This VRNN architecture shows promising results by itself but generally struggles with long-term effects which inherently can be a problem for RNN based architectures. For this reason an attention mechanism is introduced to allow long-term effects to be modeled more easily. The attention mechanism introduced will allow a latent space to be attended to by less recent information and potentially improve detection on longer trajectories. The goal is to combine the ability of RNN architectures to capture short-term information and combine this with long-term information through attention mechanisms. As the goal is to transform a sequence of latent spaces into another sequence of latent spaces only a transformer encoder is needed and thus not the entire architecture. A standard example of such a transformer encoder can be seen in figure 3.5

This transformer encoder will be applied to the sequence of outputs that will be accumulated by the RNN block. The transformer will initially receive a single input from the RNN that at each timestep will grow in size. The structure of inputs to the transformer at each timestep will thus change slightly by expanding along a single dimension.

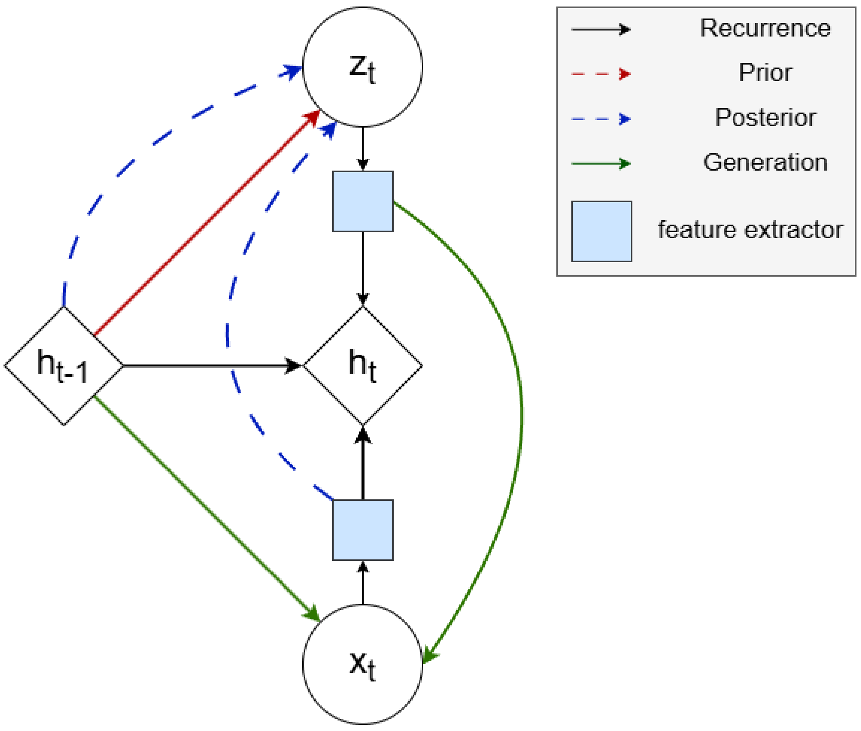


Figure 3.4: The newly added blue squares represent a non-linear function which in this is an MLP which works as a feature extractor for inputs and interpretation of latent space. Basic schematic is borrows from original VRNN authors [10]

```
#input sequence at timestep t
Input_1 = [Z_1]                   #t = 1
Input_2 = [Z_1, Z_2]              #t = 2
Input_3 = [Z_1, Z_2, Z_3]         #t = 3
...
Input_n = [Z_1, Z_2, ..., Z_n]    #t = n
```

With each Z representing a latent space in a sequence of length n. This will result in inputs reinforcing the memory of the model and thus alleviating the tendency for RNN architectures to "forget" previous states. With this extension the architecture which will be evaluated and compared to the original purely VRNN based GeoTrackNet implementation can be seen in figure 3.6

Both models operate much the same and can thus share a training loop providing the same data and processing the same outputs. In order to start the training process the 4-Hot encoded inputs are fed into the models Which is then passed forward where the outputs are a collection of probabilities being :

- $p(z|h_{t-1})$: being the prior.
- $q(z|x,h_{t-1})$: being the approximate posterior.
- $p(x|z,h_{t-1})$: being the generative output

The loss function where these outputs are utilized for this architecture is the Evidence Lower BOund (ELBO) loss. This loss is central to training variational models like VRNNs and normal VAE models. The ELBO training objective function can be seen in equation 3.1

$$\mathscr{L}_{\text{ELBO},t} = \underbrace{\mathbb{E}_{q(z_t|x_t,h_{t-1})}\left[\log p(x_t \mid z_t, h_{t-1})\right]}_{\text{(1) Reconstruction Term}} - \underbrace{\text{KL}\left[q(z_t \mid x_t, h_{t-1}) \,\|\, p(z_t \mid h_{t-1})\right]}_{\text{(2) KL Divergence Term}} \tag{3.1}$$

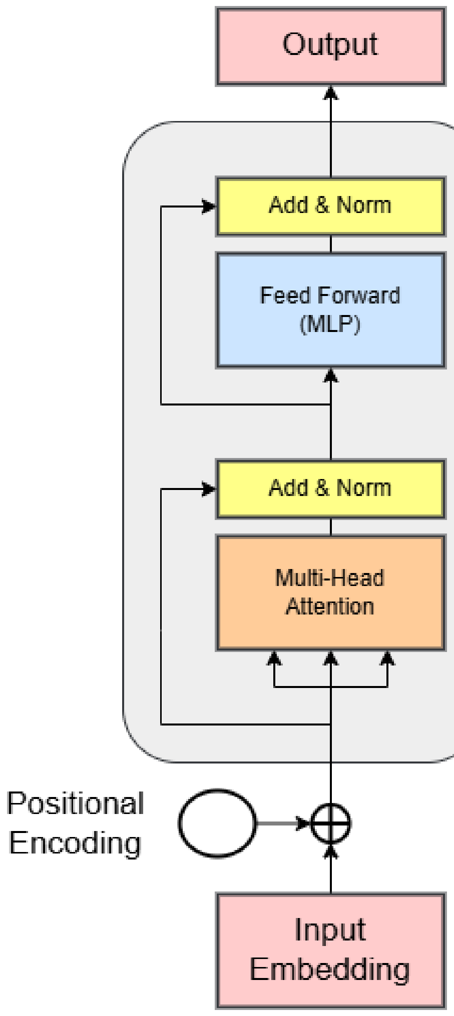


Figure 3.5: Illustration of standard transformer encoder as introduced by Vaswani et al. in ”attention is all you need” [12], figure is also a recreation of their original schematic.

As visible in equation 3.1 the ELBO loss i constructed from two separate terms being the reconstruction term and the Kullback-Leibler (KL) divergence term.

The reconstruction term aims to maximize the expectation from the reconstruction and thus maximize the probability. This is only possible with the assumption that the training data is considered normal and thus a naive maximization of the probability given the input sample is possible.

The KL divergence term aims to minimize the difference between the prior and the approximate posterior in order to provide a stronger understanding of how normal data is distributed before a sample has been passed through to create a new approximate posterior.

Evaluating the ELBO loss in its entirety with the terms explained, it is important to realise that ELBO loss is designed for maximization as suggested by the subtraction between the terms. A high generative probability is good and a low difference in prior and approximate posterior is good thus resulting in a large loss, when a small number is subtracted by a large number. Thus for training purposes the actual loss used is the negative ELBO loss.

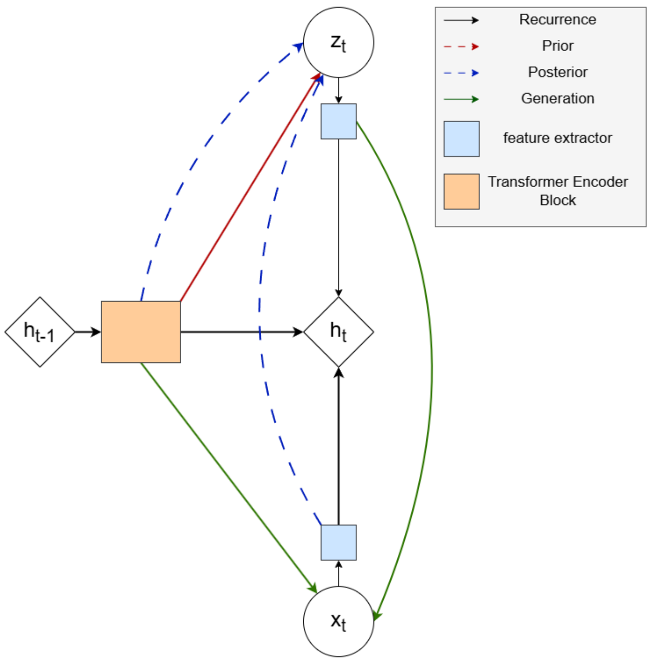


Figure 3.6: The GeoTrackNet VRNN with added transformer encoder for improved long-term modeling. Here the transformer is placed right after each reccurence in order to transform outputs based on attention mechanisms.

In practice this loss is accumulated across samples in a track and is thus divided out by sequence length resulting in a final ELBO loss which can be seen in equation 3.2

$$\mathscr{L}_{\text{ELBO}} = \sum_{t=1}^{T} \frac{1}{seq\_length} \mathscr{L}_{\text{ELBO},t} \tag{3.2}$$

The division by track length doesn't provide more information but reduces the magnitude of the loss and thus provides a more stable training process.

3.3 Final Detection

With the final output of the model acquired the final stage of detection can be performed. The model itself is used as a way to evaluate normalcy as it builds on the idea of an autoencoder meaning that data similar to what was trained on will be reconstructed to a satisfactory degree whereas dissimilar data will be reconstructed to a lesser satisfactory degree. This means that with the model modeling normal behavior a low probability output will result in a potential anomaly. With this a potential global threshold could be defined to mark a sample as anomalous. This however is not ideal due to the varied density of data and the variability of said data. In certain areas, vessels might perform more predictable actions than in other areas. In open water it is likely cargo ships sailing along well-defined shipping lanes whereas near ports a more chaotic or unpredictable movement pattern might be present. This would skew the anomaly detection towards difficult to identify areas and not directly highlight actual anomalous behavior.

For this purpose a contrario detection will be used in the form of geospatial a contrario detection which will divide the ROI into cells much like the encoding for the model prepared for. This allows individual thresholds to be defined within each cell and thus thresholds are indirectly local to an area instead of global. This is done in a few steps and allows for single anomalous AIS messages to be ignore as they are likely standalone anomalous events unlikely to represent general anomalous behavior.

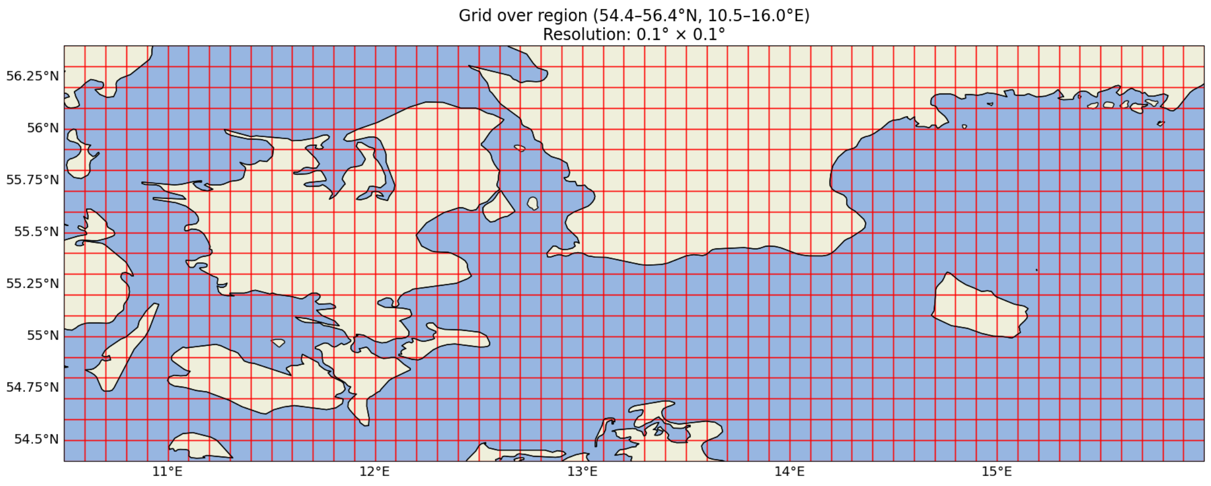


Figure 3.7: A map of the ROI with a highlighted cell grid highlighted with red lines. The resolution is ten times smaller than what is used during calculations, for visual clarity.

To make the explanation of the method easier it is assumed that the entire testset has been passed through the model and provided a series of outputs functioning as a proxy for the log-likelihood. This means that each sample in each trajectory will have a predicted log-likelihood attached to it. These log-likelihoods will then, depending on location, be assigned to a cell in the defined grid from the ROI. An example of this grid structure of the ROI is shown in figure 3.7.

Within each of these cells a gaussian kernel density estimation is performed in order to estimate the parameters of probability distributions within each cell. This is the fundamental idea that allows each cell to contain its own relative threshold instead of applying a single global threshold.

Passing a sample through the model will then grant a log-likelihood which can now be compared to the distribution inside the cell matching geographical location of the input. This is done by taking the cumulative density function (CDF) of the estimated gaussian and checking if it is below a certain value "p". This concept is formalized in equation 3.3

$$\Phi\left(\frac{\mathscr{L}-\mu}{\sigma}\right) < p \tag{3.3}$$

Here the "$\mathscr{L}$" refers to the log-likelihood where "μ" and "σ" refer to the parameters of the gaussian distribution within the local cell. Now every sample can be marked as potentially anomalous which is in itself a way to detect tracks as anomalies under the assumption a single anomalous sample makes a track anomalous. This is however not a very robust approach as singular anomalous messages is possible and would quickly spike the rate of false positives. Instead a few extra steps are taken where tracks are now considered in segments.

When each of these samples in a trajectory has been analyzed and assigned wether or not they are possible anomalies. This is equivalent to forming a list assigning 1 to samples which are possible anomalies and 0 to the rest. With a list containing potentially anomalous points, it is possible to measure the probability that at least k out of n AIS samples in an AIS trajectory with length n of this track is marked as an anomaly. This is the tail of a binomial distribution $P(X > k)$ describing the probability of an event occurring at least k times. The purpose is seen with the binomial distribution shown in equation 3.4 and then the tail of the binomial shown in equation 3.5.

$$P(X = k) = \binom{n}{k} p^k (1-p)^{n-k} \tag{3.4}$$

$$P(X \geq k) = B(n,k,p = \sum_{i=k}^{n} \binom{n}{i} p^i (1-p)^{n-i} \tag{3.5}$$

With the analysis of this binomial tail distribution, a contrario detection bases its detection of anomalies on the Number of False Alarms (NFA) which is defined as in equation 3.6

$$NFA(n,k,p) = N_s B(n,k,p) \tag{3.6}$$

Where $N_s = \frac{T(T+1)}{2}$ is the number of all possible segments of a trajectory $x_{1:T}$ made up of samples x_t. An example of this being $T = 3$ resulting in 6 possible segments. 3 segments of length 1, 2 segments of length 2, and 1 segment of length 3. This finalizes the definition of geospatial a contrario detection where only a final comparison to a global threshold is defined. This threshold is denoted ε and adjusts the amount of expected false positives when applied to a normal set. This value is usually defined as a really low value but can vary based on how strict the detector needs to be. An example would be to focus on a cell in the grid created from the ROI. If a sample from this set is drawn $\frac{1}{\varepsilon}$ times the result would be expected to result in 1 false alarm. This means an $\varepsilon = 0.001$ would result in a single false alarm every 1000 measurements of normal data. Because the global threshold is measured against a localized likelihood measurement, the global threshold is now tied to number of allowed false alarms instead of a direct cutoff that will return the same result regardless of geographical location.

4 Evaluation

This chapter will present a short section describing the experimental setup with information such as batch size, learning rate, dimensions, and other hyperparameters. This will together with the model architecture descriptions allow for reproducibility. Afterwards the results will be presented in the form of graphs, tables, and matrices presenting the results with matching descriptions and explanations of what they show. Through these results an answer to the problem statement can be concluded with nuances further explored within chapter 5

4.1 Experimental Setup and Hyperparameters

Throughout this section a thorough presentation and explanation of parameters and their respective values will be presented as most of them can have a major influence on results and performance. The parameters will be sorted into three subsections will will explain parameters within the dataset, the models, and at last the detection algorithm.

Dataset parameters

As discussed in chapter 3 the dataset was preprocessed in a way that allows for the categorical 4-Hot encoding that is inputted to the model initially. In order to replicate this a few parameters need to be defined. As the cell is divided by the ROI the input features to the model vary based on both the size of this ROI and the resolution with which the 1-Hot encoding is performed on the latitude and longitude. A similar processing is applied to speed and course and thus a similar resolution is defined for those. With this in mind the parameters which are important and their parameters are shown in table 4.1 The

Parameter	Value
Minimum latitude	54.4 decimal degrees
Maximum latitude	56.4 decimal degrees
Minimum longitude	10.5 decimal degrees
Maximum longitude	16.0 decimal degrees
Latitude encoding resolution	0.01 degrees/cell
Longitude encoding resolution	0.01 degrees/cell
Maximum SOG	30 knots
SOG encoding resolution	1 knot/bin
COG encoding resolution	0.5 degrees/bin

Table 4.1: Caption

most interesting and hard to interpret values are the latitudes and longitudes. These are in decimal degrees and thus change meaning based on where on the globe they are defined. These parameters decide the size of the input being calculated as shown in equation 4.1 with Δ referring to the resolution of the respective parameter.

$$N_{\mathrm{lat}} = \left\lceil \frac{\mathrm{lat}_{\max} - \mathrm{lat}_{\min}}{\Delta_{\mathrm{lat}}} \right\rceil, N_{\mathrm{lon}} = \left\lceil \frac{\mathrm{lon}_{\max} - \mathrm{lon}_{\min}}{\Delta_{\mathrm{lon}}} \right\rceil, N_{\mathrm{sog}} = \left\lceil \frac{SOG_{\max}}{\Delta_{\mathrm{sog}}} \right\rceil, N_{\mathrm{cog}} = \left\lceil \frac{360^\circ}{\Delta_{\mathrm{cog}}} \right\rceil \tag{4.1}$$

Equation 4.1 clearly shows how the size of the ROI and the decided resolution affects the size of the input to the model. With the values defined in table 4.1 a total of 4 vectors would be created with lenghts :

$$N_{\text{lat}} = 200, \quad N_{\text{lon}} = 550, \quad N_{\text{sog}} = 30, \quad N_{\text{cog}} = 72$$

$$N_{total} = N_{\text{lat}} + N_{\text{lon}} + N_{\text{sog}} + N_{\text{cog}} = 852$$

This means that training and thus inference is done with an input of 852 values where four values are one and the rest are zeros.

Model parameters

It is important to be transparent with parameters when comparing models as they can have a major impact on visible performance. The results later in the chapter will present results from two different models with comparable but still different architecture. The introduction of an attention mechanism poses a set of challenges that requires intentional and precise tuning of hyperparameters. A quick overview of all relevant hyperparameters can be seen in table 4.2.

hyperparameter	GeoTrackNet	Attention extension
Batch size	32	2
Learning Rate	0.0003	0.0001
Hidden state size	120	120
Latent space size	120	120

Table 4.2: Caption

A further set of hyperparameters introduced as a result of the transformer encoder means that the attention extension adds a few extra hyperparameters to adjust. These parameters are amount of layers, number of attention heads, and model dimension. The have been assigned the following values for the training session that is evaluated through this chapter :

- Attention heads : 2
- Model dimension : 120
- Number of layers : 2

The original model contains just over 550.000 parameters and with the transformer encoder block this goes up to just around 900.000 parameters. This increase is due to the the transformer architectures many quadratically scaling layers.

Detection parameters

Furthermore some hyperparameters are also defined for the detection part as it builds on some preprocessing steps before actual detection is performed. These hyperparameters can be seen in table 4.3.

Parameter	Value
Latitude cell resolution	0.01
Longitude cell resolution	0.01
p from equation 3.3	0.1
ε threshold	$[1 \cdot 10^{-5}, 1 \cdot 10^{-6}, 1 \cdot 10^{-7}, 1 \cdot 10^{-8}, 1 \cdot 10^{-9}, 1 \cdot 10^{-10}, 1 \cdot 10^{-11}]$

Table 4.3: The parameters used during definition of the detection algorithm. The resolutions defining cells in which likelihoods are compared, p from which initial anomaly candidates are defined, and ε defining a series of values for final anomaly detection used during testing

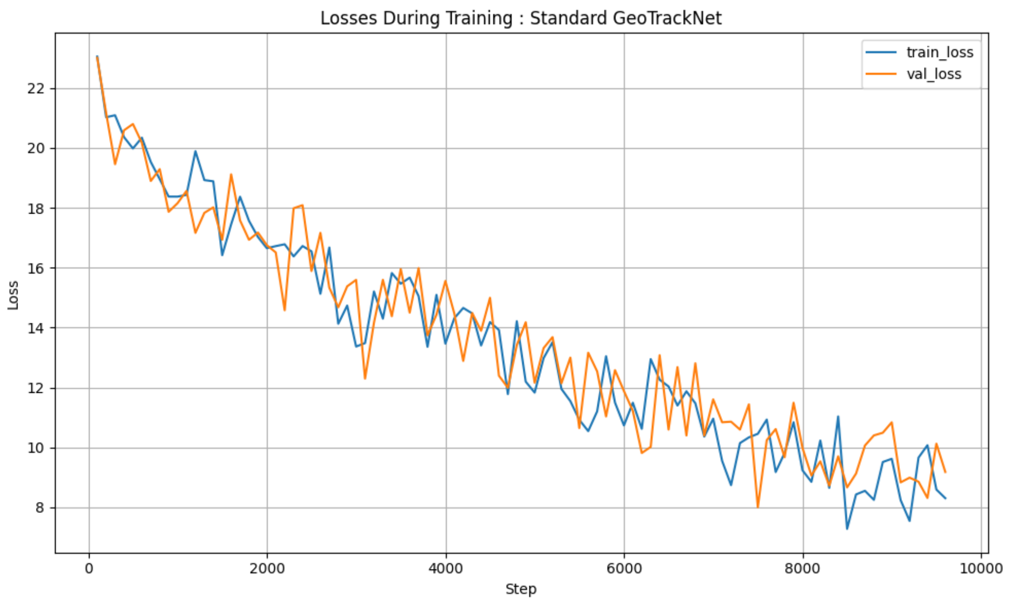
4.2 Results

This section will introduce the actual results through testing and compare the original GeoTrackNet model with transformer extension. They will be compared on both training progression as well as performance metrics with the labeled dataset allowing classification performance evaluation. A major test will be a thorough analysis of the models performance through multiple threshold values in the a contrario detection to uncover nuances in the models ability to more consistently distinguish between anomalous samples and normal samples.

Training Progression

The training process between the two models vary significantly. Due to the increased amount of parameters the original Geotracknet took 0.52 seconds per step with a batch size of 32 resulting in ≈ 0.06 seconds per sample. This is quite a bit faster compared to the model with the transformer extension as this model took 0.58 seconds per step with a batch size of 2 resulting in ≈ 1.72 seconds per sample. This is largely explained by the CPU based training as parallelization increases performance of attention to a high degree.

The loss curves for both the models can be seen in figure 4.1 showing the training loss and validation loss for both models.



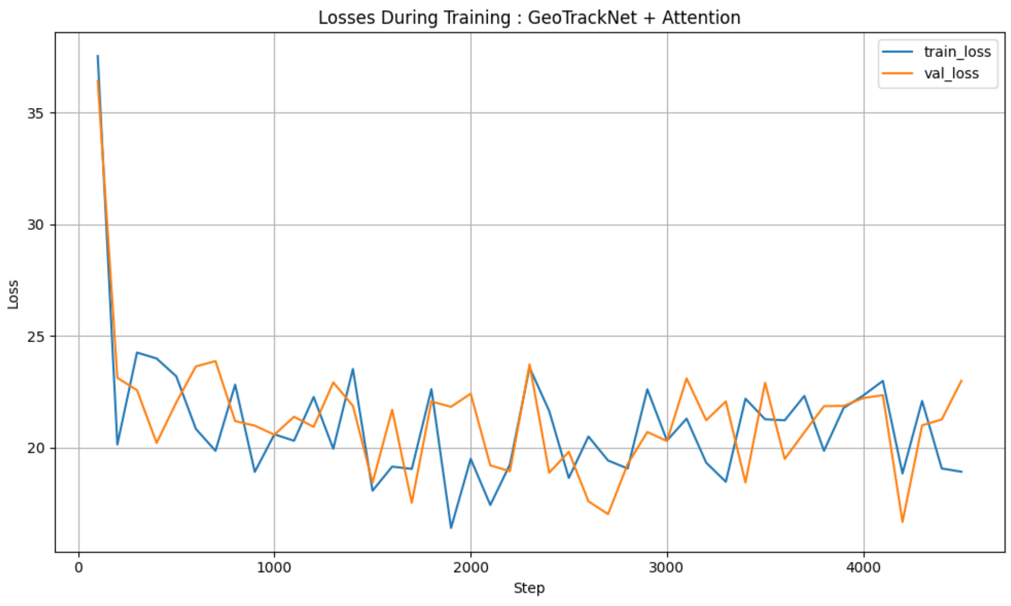


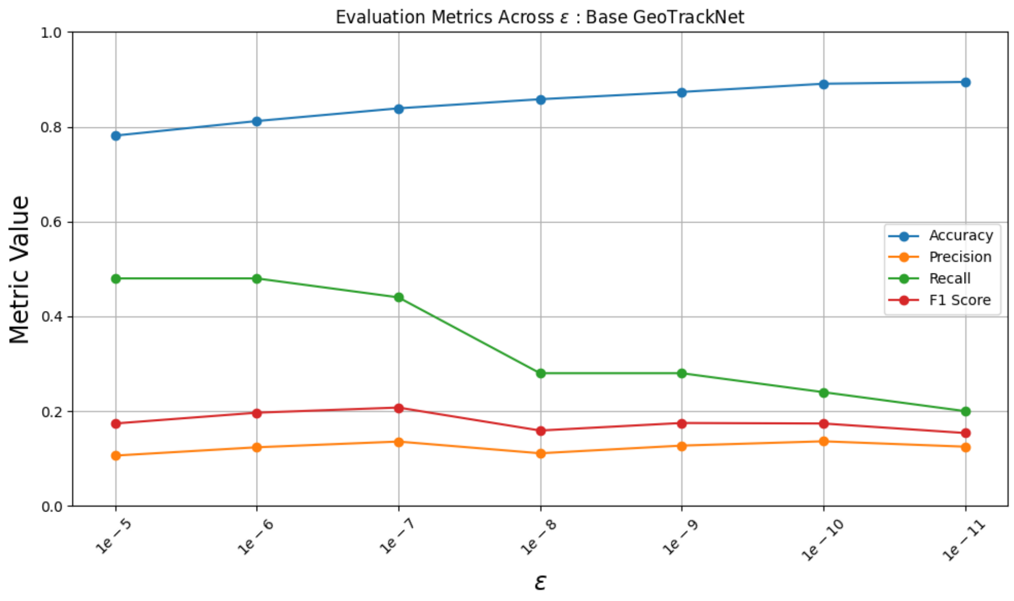
Figure 4.1: A comparison between losses during training of both the standard GeoTrackNet and the attention extension. Be aware that the initial value is not logged and starts at ≈ 560 for both models.

The loss curved between the models look very different and with good reason. The first model shows a clear sign of simply being too impatient regarding early stopping as no clear signs of overfitting has emerged.

The loss curves for the attention extension looks different and has no general obvious downwards trend strongly suggesting a potential learning rate problem. The loss curves suggest that the model learned some basic patterns in the first few steps and then stagnated without convergence.

Classification results

This section will explore the classification abilities of the models. The only parameter that will be adjusted throughout these tests is the ε as this is the final threshold defining the trade-off between false positives and false negatives. Here some differences between the models might show up and suggest situations in which one model outperforms the other. Firstly four popular metrics are evaluated across the selected range of ε values. The evolution of these values can be seen in figure 4.2 showing accuracy, precision, recall, and f1-score of both methods across ε values.



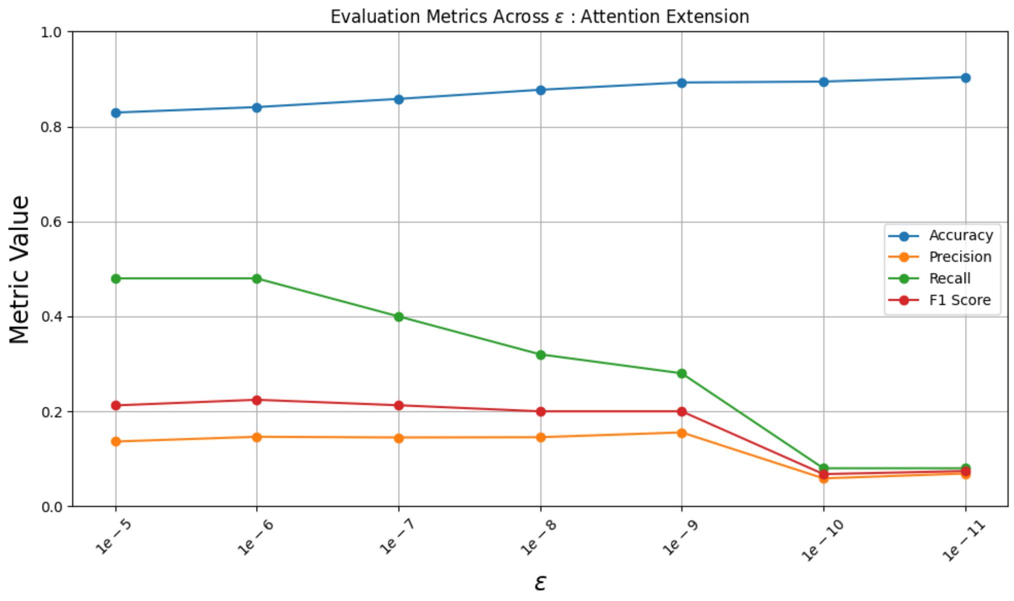


Figure 4.2: A comparison between classification metrics across both model variants is compared across a range of ε values used in the a contrario detection.

From figure 4.2 it is observed that both models experience a big drop in performance according to all metrics except Accuracy which could indicate that a class is being predicted wrongly more often after this point. Generally the models seem to perform comparably with very little difference. Due to the large class imbalance between normal and anomalous samples the f1-score is a very important metric to consider due to the focus on minority classes and thus preferable over accuracy.

Due to the low f1-score a series of confusion matrices have been produced in order to more closely monitor tendencies within classification predictions and ground truths. A series of three confusion matrices are produced for the standard GeoTrackNet corresponding to the lowest-, highest and a median ε value and i shown in figure 4.3

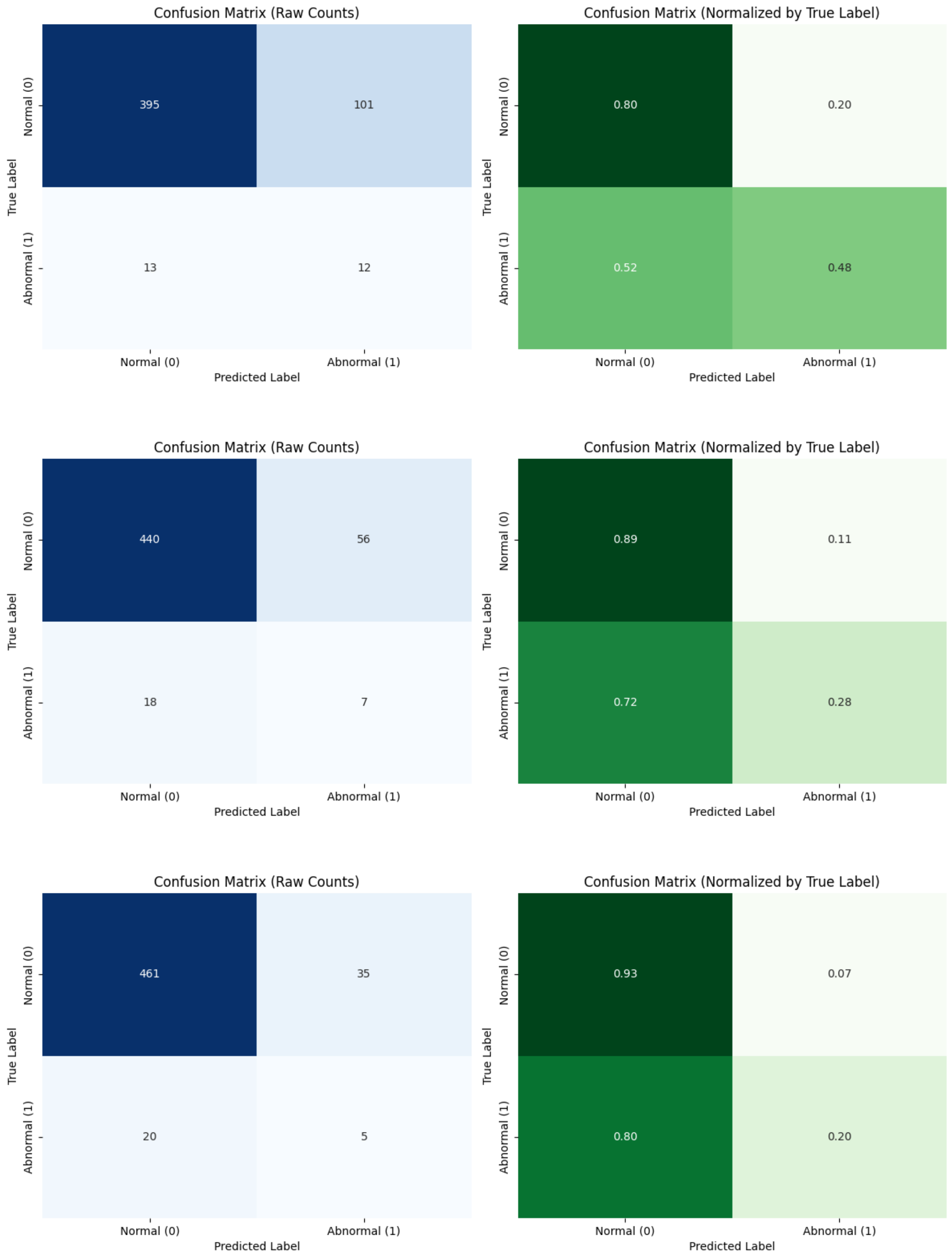


Figure 4.3: three confusion matrices showing classification from standard GeoTrackNet across three different epsilon values. top matrix has an ε = 1e-5, middle matrix has an ε = 1e-8, and the bottom matrix has an ε = 1e-11

These matrices confirm the low f1-score by showing a general tendency to classify abnormal tracks as normal. The model however seems to not struggle as much with classifying normal tracks as abnormal.

Continuing this evaluation, a similar set of confusion matrices for the attention extension can be seen in figure 4.4

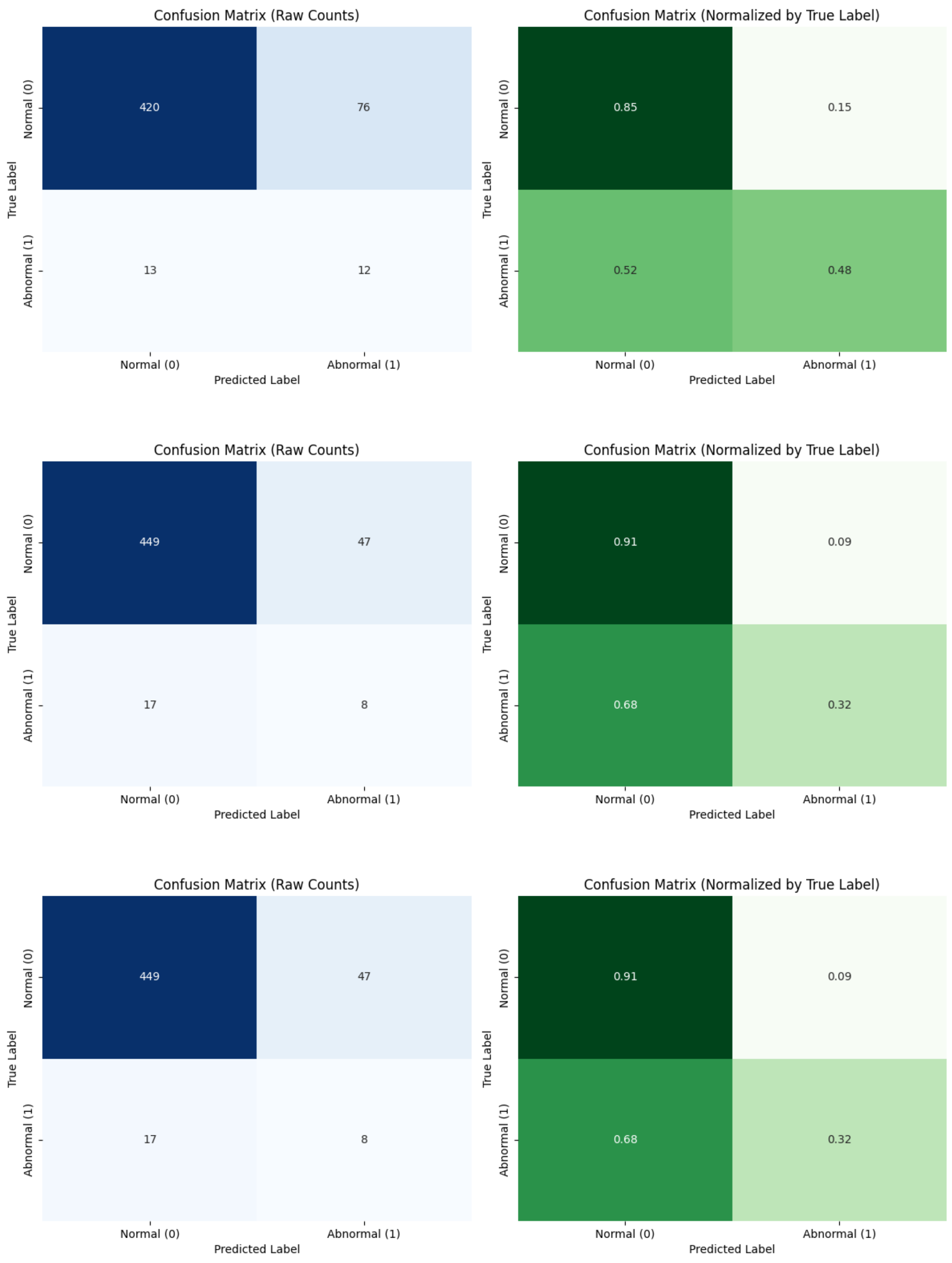


Figure 4.4: three confusion matrices showing classification from the attention extension across three different epsilon values. top matrix has an ε = 1e-5, middle matrix has an ε = 1e-8, and the bottom matrix has an ε = 1e-11

It is close to the same results but with a slight advantages in identifying anomalous tracks. This is a small but potentially significant difference. It does however in the strictest scenario with the lowest ε-values end up mistakenly classifying normal tracks as anomalous slightly more often than the standard GeoTrackNet results.

The results generally show a very strong bias towards the negative class being the normal class. Across all tested values of ε, the detection rate of normal tracks as abnormal maintains a relatively steady percentage. This however changes if the ε values is changed more drastically to allow for more false alarms. Setting ε to a value of 1e-2 being three orders of magnitude larger than the highest values in the other tests reveals some interesting results. These can be seen in figure 4.5 showing the normalized confusion matrices for both models.

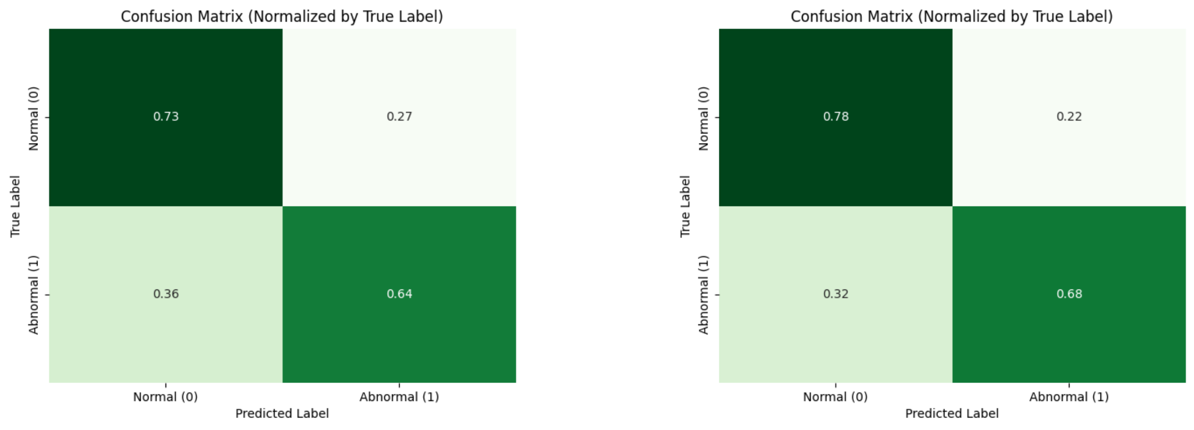


Figure 4.5: Normalized confusion matrices with Standard model on the left and attention extension on the right. The results are gathered from a contrario detection using an $\varepsilon = 0.1$

These results does not show that by simply adjusting ε, a perfect classifier can be achieved. Instead it illustrates how the decision between false positives and false negatives can be adjusted based on what is of interest and what is deemed most important.

A final visualization is produce to get an idea of how the anomalies look on actual trajectories. For this the ε which provided the most balanced distribution beween false positive and false negatives have been used and thus the trajectories seen in figure 4.6 and figure 4.7 are produced by plotting a random subset of trajectories which has been marked as anomalous with an $\varepsilon = 0.001$

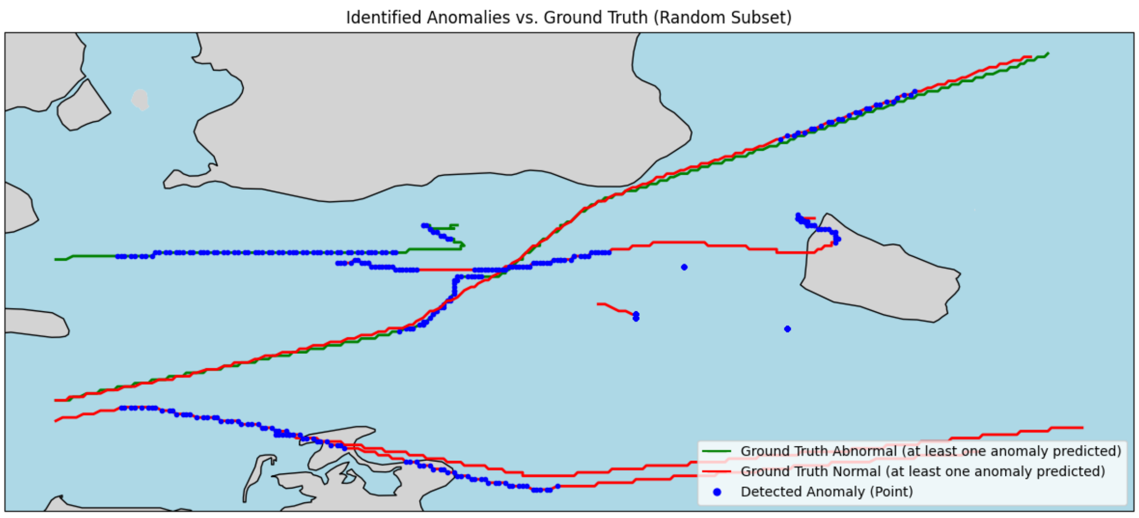

Identified Anomalies vs. Ground Truth (Random Subset)
Ground Truth Abnormal (at least one anomaly predicted)
Ground Truth Normal (at least one anomaly predicted)
Detected Anomaly (Point)

Figure 4.6: A visualization of trajectories marked as anomalous under $\varepsilon = 0.001$ using the standard GeoTrackNet, with green denoting true positives and red denoting false positives. Specific anomalous points identified are highlighted using blue dots.

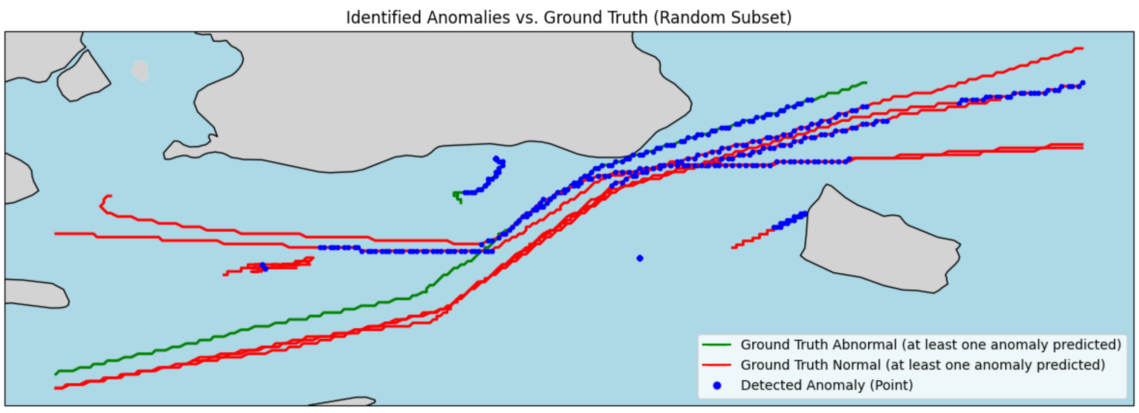

Identified Anomalies vs. Ground Truth (Random Subset)
Ground Truth Abnormal (at least one anomaly predicted)
Ground Truth Normal (at least one anomaly predicted)
Detected Anomaly (Point)

Figure 4.7: A visualization of trajectories marked as anomalous under $\varepsilon = 0.001$ using the attention extension, with green denoting true positives and red denoting false positives. Specific anomalous points identified are highlighted using blue dots.

5 Discussion and Conclusion

This chapter will contain two main sections being the discussion where a more interpretive and direct approach will be taken, where the author will express personal evaluations and thoughts about implementation, methods, and results. At last the conclusion will be drawn where the problem statement will be answered in detail with any caveats that might be important to consider.

5.1 Discussion

This discussion will have a heavy focus on challenges regarding computation and some undocumented implementation that was never used due to technical difficulties and time constraints. Generally this discussion will subjective language as the author will express their own thoughts during this section

Implementation challenges

Before discussing anything else it is important to mention for later context that the entire framework was mostly ported to Pytorch in order to use more complete implementations and to make use of previous experience in a framework. This however did not go as planned as multiple results deviated from what was expected by running the same tests in the original Tensorflow 1.x implementation. For this reason the framework that everything is tested on is implemented in Tensorflow 1.x on top of the GeoTrackNet implementation. For this reason, training and initial inference is run using Tensorflow 1.x static graphs with results being exported to external evaluation programs.

Due to lacking experience using Tensorflow 1.x and its graph execution mode a lot of implementation around training, evaluation and final inference took a lot of time and effort to accomplish. Executing code fro ma pre-built static graph functions differently than the otherwise popular eager execution which Pytorch uses and Tensorflow 2.x have also adopted. Another side effect of using Tensorflow 1.x is the specific dependencies required if GPU utilization is needed during training. This is not directly compatible with newer drivers which is required by the more moderne GPU's such as the NVidia RTX 30-series and up. For this reason some slight changes was made making it possible to execute older Cuda and Cudnn drivers from a local environment while still having the original required drivers installed for the main functionalities of the GPU. In this way it was suddenly possible to train on a GPU device. This approach however quickly turned out to be unstable and used about twice the memory otherwise used during training. For this reason the GPU training was dropped as the models did not fit on the 10GB RAM available on the GPU.

With the GPU dropped, the training times suddenly took much longer than anticipated. This is not a big problem on the standard GeoTrackNet implementation as it is quite small and effective by itself. Introducing attention mechanisms through the transformer encoder however turned out to be much more resource intensive than originally anticipated. Not being able to train on GPU also eliminates the paralelizability advantage of multi-head attention, which makes CPU training even worse.

Comments on Achieved Results

These challenges culminated in the inability to optimize hyperparameters which can make a big difference in the final result. Especially when other parameters needs to be compromised in order to make it happens. If it is not the learning rate resulting in the attention extension model to fluctuate in training loss then it is likely the low batch size resulting in loss-noise from the probabilistic and uncertain results with a small sample size for each update.

Having a model that is not trained to a certain standard can impact its performance making a lot of the results potentially skewed. It is unlikely to completely change the results that was calculated but might turn out to have a better recall than what is shown here. The interesting results is the combination of a less than ideal loss curve during training of the attention extension but the ability to compete with the standard GeoTrackNet anyways. This to me shows potential for getting strong results by performing a more ideal training session with expanded RAM such that a slightly bigger more expressive attention block can be added which should improve the modeling capabilities.

It is also worth commenting on the a contrario detection which has been performed. As mentioned in the chapter 3 it is important to have a baseline for each cell in your grid map. If this is not the case you are unable to properly evaluate wether a vessels behavior is anomalous or not. This concept is explored a bit in the original paper but does not make sense in a situation like this where the labeled dataset, from which the classification metrics have been derived, are collected around a specific region inside the ROI. As mentioned previously in chapter 2, under the related works section, the dataset only contains 25 labeled trajectories which are considered anomalous. This is a challenge when it comes to location based analysis of performance as they are all indirectly connected to the same anomaly, being a collision event resulting in coast guard vessels and alternative routes being used by other vessels. For this reason no location specific analysis have been done.

Finally, I acknowledge the GAN-based approach which was mention earlier in chapter 2. This approach i initially chose not to pursue due to concerns about the real-world feasibility of samples and their effect on performance, and the lack of physical constraints in the generation network. While this critique still stands, especially the early evaluation methodology, I also recognize that my current approach requires further work and adjustment to reach comparable levels of performance. The results shown in their original work [14] highlight the potential for generative models in the domain of anomaly detection, especially within AIS-based anomaly detection. Reproducing their results however or comparing to them directly is challenging and thus not attempted. This is because the results and training has been done in different regions of the world resulting in differences in behavior and ship types. These differences would make a head-to-head performance comparison potentially misleading and thus result in inaccurate or incomplete conclusions.

5.2 conclusion

To reiterate the problem statement, which this project set out to answer and explore is defined as such :

How well does a state-of-the-art method like GeoTrackNet perform when compared to actual labeled data? and would additions like attention mechanisms that make the foundation of the TrAISformer improve the performance of GeoTrackNet by enabling longer-term dependencies to be learned?

The problem statement can be answered in two sections :

- ”How well does a state-of-the-art method like GeoTrackNet actually perform when compared to actual labeled data?”
- ”and would additions like attention mechanisms that make the foundation of the TrAISformer improve the performance of GeoTrackNet by enabling longer-term dependencies to be learned?”

These two question will be answered in detail in the two following sections.

”How well does a state-of-the-art method like GeoTrackNet actually perform when compared to actual labeled data?”

Looking at the results of GeoTrackNet, it shows a strong performance in identifying normal tracks with relatively low probability of classifying normal tracks as anomalous. This shows a strong ability to model and understand what is considered ”normal” behavior in the maritime domain based on AIS input messages. The results does however still show that work has to be done to either improve the model itself with more parameters, perform hyperparameter optimization to improve performance, or defined a stronger set of input features for modeling anomalous behavior that might not be visible in the features used throughout this project and similar ones.

”and would additions like attention mechanisms that make the foundation of the TrAISformer improve the performance of GeoTrackNet by enabling longer-term dependencies to be learned?”

The attention extension does in its current state not outperform GeoTrackNet, making it hard to say that it improves the existing standard model. A combination of training results and performance metrics does however still show a positive trend. Even with a loss that suggests convergence will not happen, together with a loss that did not reach as low a value as the standard model. The attention extension still managed to be comparable to the standard model in performance across all tests. As transformer architectures are notorious for needing hyperparameter tuning this could indicate that the addition of an extra attention block and thus the ability to model long-term dependencies can help in modeling ”normal” behavior.
Even with a potential for superior performance given the right hyperparameters and opportunity for training. The addition of a transformer encoder extends inference time significantly which is a big factor to consider if hundreds of thousands of AIS message will need to be processed each day. Given enough computing power it should be possible, but still requires more than the standard model.

Bibliography

[1] North Atlantic Treaty Organization (NATO). *AIS (Automatic Identification System) overview*. 2021.
URL: https://shipping.nato.int/nsc/operations/news/2021/ais-automatic-identification-system-overview.

[2] United States Coast Guard. *Automatic Identification System (AIS) Overview*. n.d.
URL: https://www.navcen.uscg.gov/automatic-identification-system-overview.

[3] Danish Maritime Authority. *Order on the construction and equipment, etc. of ships, implementation of the International Convention on Safety of Life at Sea (SOLAS)*. 2019.
URL: https://www.dma.dk/Media/637740485819728425/Order%20on%20the%20construction%20and%20equipment%2C%20etc.%20of%20ships%2C%20implementation%20of%20the%20International%20Convention%20on%20Safety%20of%20Life%20at.pdf.

[4] Danish Maritime Authority. *AIS Data*. n.d.
URL: https://www.dma.dk/safety-at-sea/navigational-information/ais-data.

[5] Mathias Anneken, Yvonne Fischer, and Jürgen Beyerer. "Evaluation and comparison of anomaly detection algorithms in annotated datasets from the maritime domain". eng. In: 2015.

[6] R. Laxhammar, G. Falkman, and E. Sviestins. "Anomaly detection in sea traffic - A comparison of the Gaussian Mixture Model and the Kernel Density Estimator". eng. In: *2009 12th International Conference on Information Fusion*. IEEE, 2009, pp. 756–763. ISBN: 9780982443804.

[7] Mahdi Shaabani Mojtaba Goodarzi. *Maritime Traffic Anomaly Detection from Spatio-Temporal AIS Data*. 2019.
URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3462402.

[8] Kamran Khan, Saif Ur Rehman, Kamran Aziz, Simon Fong, and S. Sarasvady. "DBSCAN: Past, present and future". eng. In: *The Fifth International Conference on the Applications of Digital Information and Web Technologies (ICADIWT 2014)*. IEEE, 2014, pp. 232–238. ISBN: 1479922595.

[9] Duong Nguyen, Rodolphe Vadaine, Guillaume Hajduch, Rene Garello, and Ronan Fablet. "GeoTrackNet-A Maritime Anomaly Detector Using Probabilistic Neural Network Representation of AIS Tracks and A Contrario Detection". eng. In: *IEEE transactions on intelligent transportation systems* 23.6 (2022), pp. 5655–5667. ISSN: 1524-9050.

[10] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron Courville, and Yoshua Bengio. *A Recurrent Latent Variable Model for Sequential Data*. 2016. arXiv: 1506.02216 [cs.LG].
URL: https://arxiv.org/abs/1506.02216.

[11] Duong Nguyen and Ronan Fablet. "A Transformer Network With Sparse Augmented Data Representation and Cross Entropy Loss for AIS-Based Vessel Trajectory Prediction". In: *IEEE Access* 12 (2024), pp. 21596–21609. ISSN: 2169-3536. DOI: 10.1109/access.2024.3349957.
URL: http://dx.doi.org/10.1109/ACCESS.2024.3349957.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL].
URL: https://arxiv.org/abs/1706.03762.

[13] Zhaojiang Lin, Genta Indra Winata, Peng Xu, Zihan Liu, and Pascale Fung. *Variational Transformers for Diverse Response Generation*. 2020. arXiv: 2003.12738 [cs.CL].
URL: https://arxiv.org/abs/2003.12738.

[14] Maohan Liang, Lingxuan Weng, Ruobin Gao, Yan Li, and Liang Du. "Unsupervised maritime anomaly detection for intelligent situational awareness using AIS data". In: *Knowledge-Based Systems* 284 (2024), p. 111313. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2023.111313.
URL: https://www.sciencedirect.com/science/article/pii/S0950705123010614.

[15] Kristoffer Vinther Olesen. *AIS Trajectories from Danish Waters for Abnormal Behavior Detection*. DOI: 10.11583/DTU.c.6287841.v1.
URL: https://data.dtu.dk/collections/AIS_Trajectories_from_Danish_Waters_for_Abnormal_Behavior_Detection/6287841/1.

A AIS data fields explained

Field	Description
1. Timestamp	Timestamp from the AIS basestation, format: 31/12/2015 23:59:59
2. Type of mobile	Describes what type of target this message is received from (class A AIS Vessel, Class B AIS vessel, etc)
3. MMSI	MMSI number of vessel
4. Latitude	Latitude of message report (e.g. 57,8794)
5. Longitude	Longitude of message report (e.g. 17,9125)
6. Navigational status	Navigational status from AIS message if available, e.g.: 'Engaged in fishing', 'Under way using engine', mv.
7. ROT	Rate of turn from AIS message if available
8. SOG	Speed over ground from AIS message if available
9. COG	Course over ground from AIS message if available
10. Heading	Heading from AIS message if available
11. IMO	IMO number of the vessel
12. Callsign	Callsign of the vessel
13. Name	Name of the vessel
14. Ship type	Describes the AIS ship type of this vessel
15. Cargo type	Type of cargo from the AIS message
16. Width	Width of the vessel
17. Length	Length of the vessel
18. Type of position fixing device	Type of positional fixing device from the AIS message
19. Draught	Draught field from AIS message
20. Destination	Destination from AIS message
21. ETA	Estimated Time of Arrival, if available
22. Data source type	Data source type, e.g. AIS
23. Size A	Length from GPS to the bow
24. Size B	Length from GPS to the stern
25. Size C	Length from GPS to starboard side
26. Size D	Length from GPS to port side