MARITIME DATA OPEN ACCESS AND ANALYTIC

by

ANDI WAHYU MULTAZAM
(B.Sc., Bandung Institute of Technology)

A THESIS SUBMITTED FOR THE DEGREE OF

MASTER OF COMPUTING

in

COMPUTER SCIENCE

in the

GRADUATE DIVISION

of the

NATIONAL UNIVERSITY OF SINGAPORE

2020

Supervisor:
Professor Stephane Bressan

Examiners:
Professor Tan Kian Lee

Declaration

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Andi Wahyu Multazam

18 April 2020

To Zia, my baby girl, I can't wait to see you soon

Acknowledgments

First, I would like to express my gratitude to my supervisor Professor Stephane Bressan for his patience, encouragement, and advise throughout the project which is surely useful for my life afterwards. This project would not have been possible without his help and guidance.

Besides my supervisor, many thanks are given to my mentor, Remmy Zein, for a great discussion and guidance. Credit is also given to Thomas Kister, the main developer behind the web liancheng.data.science whom I get insight a lot from about the AIS end-to-end system.

At last, I would like to thank my wife for her support and love during the juggle between work and study. This project would have been more difficult without her assistance. You and our incoming baby are the reason that keep me motivated to complete this project.

Contents

Abstract

Maritime Data Open Access and Analytic
by
Andi Wahyu Multazam
Master of Computing in Computer Science
National University of Singapore

With the current position as a leading maritime capital in the world, security is becoming a top priority for Singapore to develop analytic technology and intelligence in the maritime sector. The application varies from vessel tracking, trajectory prediction, to vessel monitoring and surveillance. Automatic Identification System (AIS) enables us to track vessel movement by receiving their records through a transponder device. The work in this project focuses on two parts; first to develop a standard exploratory data analysis for movement data. This is achieved by utilizing some of the techniques and tools specifically built to handle trajectory data. Secondly, to develop a prediction model to understand the future movement of the vessel given the history of AIS information. A trajectory prediction model based on Long Short Term Memory is designed, implemented, and tested on 1-month AIS data. The result suggests that the LSTM model works best at predicting 30 timestep into the future by using recursive multi-step prediction methods that utilize a shorter timestep model.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Singapore maintains its position as the leading maritime capital in the world for being tops in three pillars including Shipping, Ports and Logistics, Attractiveness and Competitiveness, and for the remaining pillars, it is within the top 10 cities [4]. The award has been received for four consecutive times by The Republic since 2012 when it was first published [26]. As Singapore continues to become one of the world's busiest port in the world, the Maritime Port Authority (MPA) needs to enhance maritime security at Singapore water against dangerous adversaries such as piracy attacks, armed robbery, crew abduction, or smuggling. The Regional Cooperation Agreement on Combating Piracy and Armed Robbery against Ships in Asia (ReCAAP) recorded the number of sea attacks in Singapore Strait jumped from 8 incidences in 2018 to 31 incidents in 2019. Because of the increase of incidents, ReCAAP recommends to the law enforcement agencies to enhance surveillance, increase patrols and respond promptly to incidents reported by ships while ships master and crew are advised to exercise enhanced vigilance when transiting the area of concern [21].

The challenges of sea surveillance in Singapore water are due to its size and vessel movements traffic. The Singapore Straits is extended for 105-km long from the Strait of Malacca in the west to the South China Sea in the east, and 16-km wide lies between Singapore Island in the north to Riau Island of Indonesia in the south. Nearly 100,000 vessels pass through the 105-km long waterway each

year, accounting for about a quarter of the world's trade good [28]. Our system records the average of 1 million vessels message in a day, leading to about 800 vessels information to keep track per minute. Those number makes it challenging for coastal police guard to patrol the vessels only when the accident takes place, hence there is a need of technology to predict the future location of vessels based on their history of movements. Some predictions would allow the police to take necessary action in advance and prevent the incidence from happening.

A standard way of sending information about ship movement is through the Automatic Identification System (AIS). Since 2002, International Maritime Organization required every ship of over 300 gross tonnages engaged in an international voyage, and passenger vessels irrespective of their size to be installed with AIS transceivers that allow a vessel to broadcast information automatically about their location in the sea. The AIS message that can be received by a nearby station equipped with AIS receiver module varies from every 10 seconds to 6 minutes.

The primary objectives of this thesis are 2 folds; first is to understand the statistics of AIS data, extract some patterns, and gain valuable insight from it. Secondly, we examined several techniques of vessel trajectory prediction using machine learning and historical data. We focus our study on AIS data of vessels crossing the Singapore Straits in December 2019.

1.2 Research Objectives

In this thesis, we seek to answer the following research questions:

- Can we gain a better understanding of AIS big data using statistics and visualization by exploiting some of their properties such as coordinates, speed, bearing, vessel ID, vessel country origin, vessel type, and destination over time?
- Can we predict the future movement of vessels based upon AIS historic information? How can we propose a prediction model to do so?

1.3 Thesis Synopsis

The rest of this thesis is organized as follows. In Chapter 2, we explain about AIS message in detail, conduct a literature review related to past works in trajectory analysis in different domain and maritime fields using AIS data, explain our end-to-end architecture system behind AIS data generation, review the state of the art of artificial neural network for sequential data. Chapter 3 describes the AIS data from a statistical point of view with visualization. We discuss AIS features and the message it entails, perform some trajectory analysis, and identify erroneous values across all features. Chapter 4 provides a methodology of designing prediction models, data pipeline, data cleaning, feature selection, and the rationale behind model hyper-parameter choice. In Chapter 5 we design and implement an experiment to find out under what condition the prediction model can suggest a better performance. We conclude the entire thesis as well as discuss further directions for future research in Chapter 6.

Chapter 2

Literature Review

In this chapter, we introduce the literature support used for this project. To start with, we first review some of the latest studies about vessel trajectory prediction. We further explain in more detail about the AIS message and the underlying end-to-end system behind data collection and generation of the AIS data. We also review the theoretical work in the field of recurrent neural network and one of its archetypes for handling sequential data.

2.1 Trajectory Prediction Model

This section will review some of the latest works on the trajectory prediction model using the AIS dataset. The trajectory prediction model is essentially a model to predict the future movement of a vessel given its historic navigation information.

Hexeberg *et al.* [9] uses the nearest neighbor search method to predict the future of a vessel's course and speed based on historical AIS data. Nearest Neighbors Search seeks to optimize the closest neighbors for a given coordinate measured with Harvesine Distance metrics. The algorithm shows good potential for vessel trajectory prediction up to about 30 minutes ahead and can follow paths of various curvatures. However, when the trajectory is too tight in the event of branching or turning, the predicted position fails to follow the same path. This could be fixed by reducing the step length and time step duration.

Young [29] builds a random forest and neural network model to predict the future position of a vessel at a given timestep based on the clustered routes of AIS data. He clusters similar trajectories of the vessel and uses collective information about a route from all ships belong to the same cluster to determine the next position of the

vessel. He found out that random forest outperforms neural network in all regions in datasets, that random forest shows better accuracy, simpler and faster to run than the neural network.

Liraz [16] develops vessel trajectory prediction models with a Recurrent Neural Network (RNN) architecture called Long Short Term Memory (LSTM). She uses LSTM to learn the Spatio-temporal dependency of AIS data and conclude that the best way to predict vessel location is to build a model with a fixed multi timestep interval into the future as opposed to short single time step forward prediction. Neural network models in general work best with adding more data and tuning some of their hyper-parameters.

2.2 Automatic Identification System (AIS)

AIS is a transponder system designed to be capable of exchanging information between ships and between ships to coastal authorities. In 2004, the International Maritime Organization enacted Regulation 19 of SOLAS Chapter V about ship-borne navigational systems and equipment. The regulation requires every ship of 300 gross tonnages upwards and engaged in international voyage or passenger vessel irrespective of its size to be installed with AIS devices on board. Ships that equipped with the AIS transceiver device shall transmit the following information in 3 categories 1). fixed information, including MMSI number, IMO number, call sign, and type of ship. 2). dynamic information, including ship's position, position time stamps, course over ground (COG), speed over ground (SOG), and heading direction. 3). voyage information, including ship's draught, hazardous cargo, destination and estimated time arrival (ETA), and route plan. AIS messages are broadcast by transponders on ships and stations by Very High-Frequency radio (VHF) periodically. AIS messages are received by VHF receivers on other ships or stations within the frequency band range. The typical AIS message is transmitted in a form as shown in Table 2.1:

Table 2.1: Encoded version of AIS message

Type	Message
Encoded	!AIVDM,1,1,,B,B8HsgOP001ne@DP;uP@03wnTkP06,0*5F

2.3 AIS Data Collection System

The School of Computing at the National University of Singapore designed and implemented an end-to-end solution for AIS data collection, cleaning, and generation and made them publicly available in `www.data.liancheng.science`. The complete apparatus of the system consists of 5 parts:

- VHF antenna to receive the AIS message in binary format.
- GPS antenna to receive ship's current position.
- Devices called AIS receiver connected to both antennas.
- Web server called "ais-receiver" responsible for displaying the ship on a map in real-time.
- Web server called "neptune" responsible for publishing the decoded AIS message logs in JavaScript Object Notation (JSON) format every month.

At the beginning of the process, AIS message is received by the AIS receiver device connected to both antennas in binary/encoded format as shown in Table 2.1. The next machine connected to the first receiver via USB is responsible for performing 3 tasks. *First*, reading the message from the first receiver through a C++ program that reads the stream of the message, logs them to the hard drive, and dispatch them in real-time over the network. *Second*, monitoring the ships via a service called "navigator", a small light Sinatra web framework written in Ruby language to display the ship's navigation using Google Map in real-time. It decodes information necessary to display the location on the map such as longitude, latitude, course, and type of vessel. Currently "navigator" uses NUS Open ID for authentication, which means it can only be accessed with NUS account. For future development, "navigator" will be set up with Nginx as a reverse proxy for better concurrency performance. *Third*, archiving the message logs via a service called "cron", a Ruby script run once a month which does a set of actions for the AIS message logs from history. The "cron" service essentially perform following actions:

- archives the AIS binary logs, which typically consist of the starting time of the record, GPS location, single-line message, multiple line messages, invalid messages,
- pre-process the binary logs by filtering invalid messages and noise, and output the result in JSON format,
- count the messages in the archives and updates the ship's database,
- store the ship information into open source database platform PostgreSQL on a yearly basis.

The last machine in the system is called "neptune", a web service that connects to "ais-machine" through TCP/IP socket such that allows them to access all archives of AIS message and make them available on the web in `www.data.liancheng.science`. The web page is refreshed once a month by running a ruby script that copies all archives from "ais-machine" to "neptune" and generating a new static HTML to display the new archived files.

2.4 Artificial Neural Network

Artificial Neural Network has recently become a powerful tool to perform machine-learning related tasks using image, text, and audio data. Artificial Neural Network is a model computation inspired by the structured of neural networks in the brain. In a much-simplified version of the brain, it consists of many computing devices (neuron) that are interconnected, through which the brain can carry out highly complex computation [25]. A simplified version of neuron is called *perceptron*, and we will discuss the mechanics of perceptron to understand more complex neural network. Perceptrons were developed in the 1950s by scientist Frank Rosenbaltt. A perceptron takes multiple binary input and produce a single binary output. Suppose perceptrons takes several input x_1, x_2, and x_3 and output y. Rosenbaltt propose a simple rule to compute the output by introducing *weights* denoted as w_1, w_2, ..., a real numbers expressing the importance of each input; the higher the value, the more important the input is [18]. The output y is computed by multiplying each input with the corresponding weight. Mathematically it is expressed as $y = w_1\ x_1 +$

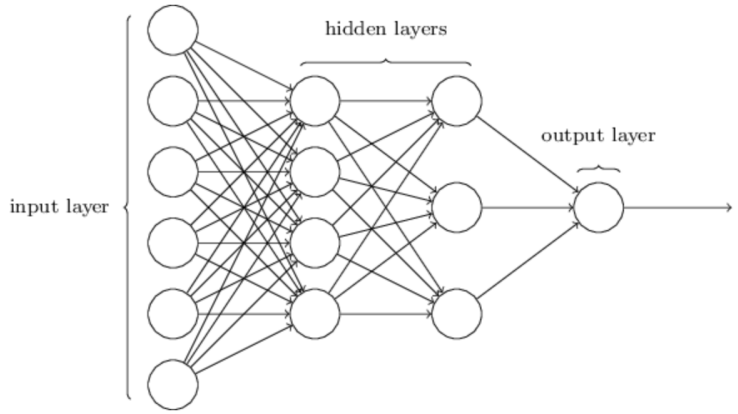


Figure 2.1: Multi layer perceptron with 4 layers and 2 hidden layers. Picture taken from Neural Networks and Deep Learning, Michael Nielson

$w_2\ x_2 + ... + w_i\ x_i$. The perceptron output is either 0 or 1 depending on whether the output y is less or greater than certain tresshold. Perceptrons "know" the value of its *weights* from fitting the input, or sometimes referred as training. Usually they do not get the most accurate *weights* in the first shot as it starts-off with random value. As the perceptrons learn the pattern of input and output, eventually they get the *weights* approximately correct. Coming back to discussion of neuron, a nueron is simply a version of perceptron with an activation function in its output. Activation function, just like any function, defines the output given its input.

From the single perceptron, we can build a larger networks consist of multiple perceptrons and even multiple layers. In fact, there exist such network named Multiple Layers Perceptrons (MLP). The term "layers" literally means the layer of network whereby the first layer takes the input from data, the second layer takes the output from the first layer as the input, and so on, until the last layer can only produce an output. The following MLP in Figure 2.1 has 4 layers with two of them are hidden layers. We can add more layers and neurons depending on how complex the data is. A common practice would be to start building a simple network with small layers and see the performance. Then we can decide to add or even reduce the network complexity accordingly.

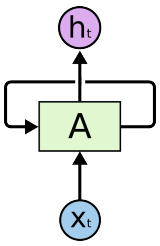


Figure 2.2: RNN network with internal loop. Picture taken from Chris Olah

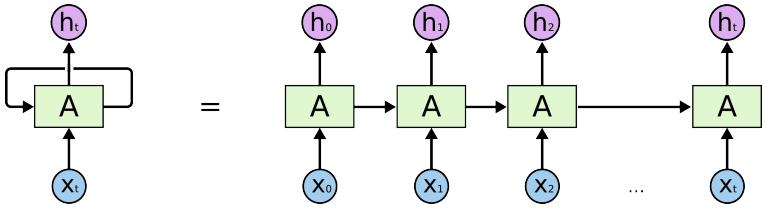


Figure 2.3: RNN network processing sequence of input. Picture taken from Chris Olah

2.4.1 Recurrent Neural Network (RNN)

The standard neural network we have discussed previously failed to address the fact that human brain process a sequence information like sentence by understanding the previous words or even previous sentences. Sequence data could be text or any data that have time dependency. In other words, there is need for another version of neural network to remember the context of previous sequence while iterating through the next sequence. Recurrent Neural Network exactly address this issue. RNN is a network with internal loop allowing the information to persist. The network is presented in Figure 2.2.

Figure 2.3 is how the RNN process the sequence of input represented by x_0, x_1, x_2, ... x_t and have output of y_0, y_1, y_2, ... y_t. The network process sequence data by iterating thorough the sequence elements and maintaining a *state* containing information relative to what it has seen so far [2]. RNN has been useful to solve many problems including language modelling, image captioning, and translation. However there is problem with the performance. RNN have difficulty in remembering information from a long time ago. It could be failure to remember specific information in the first sentence of the paragraph while processing the last sentence. RNN is also known to be computationally expensive.

Long Short Term Memory (LSTM) come to solve the long term dependency

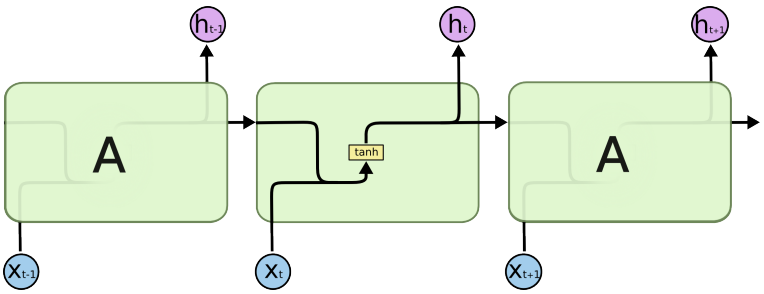


Figure 2.4: Standard RNN internal mechanichsm. Picture taken from Chris Olah

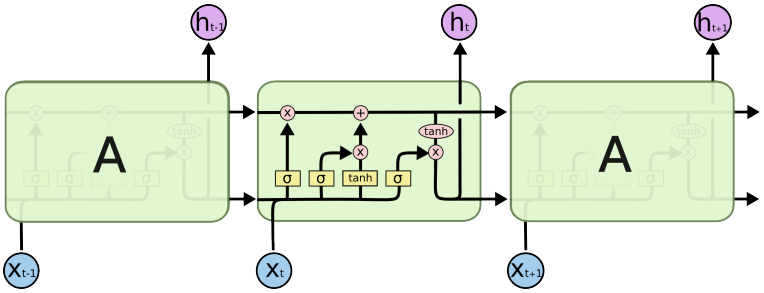


Figure 2.5: LSTM, an improved version of RNN introduce new node called gates. Picture taken from Chris Olah

problem that RNN have a hard time with in practice. LSTM was introduced first time by Hochreiter & Schmidhuber in 1997, and improved over the years by many studies. LSTM is specifically designed to remember long term information by introducing new node called *gates*. The core principle of LSTM is still similar to the standard RNN, they are just different in the internal mechanics of single network.

There we see in Figure 2.5, the *gates* node was introduced with sigmoid function and pointwise multiplication. The purpose of the gates is to decide which information to "forget" or pass depending on the output of sigmoid activation function. A value of zero means "let nothing through" and a value of one means "let everything through" [20]. The gate act like information regulator which evaluate the importance of information to let through to the next cell.

Chapter 3

Exploratory Data Analysis

The following chapter discusses exploratory and analysis of AIS data using statistics and visualization. To start off, we first explain about AIS features description. We continue with AIS data exploration using statistics and visualization, which include erroneous value identification, and trajectory overview.

3.1 Data Description

Looking back to the Introduction chapter, the typical AIS data packet is encoded as follows: `!AIVDM,1,1,,B,177KQJ5000G?tO`K>RA1wUbN0TKH,0*5C`. Each field is separated by a comma as there are 7 fields in the packet. Field 1 to 5 together with field 7 are specific to the encoding configuration of the data packet to enclose the actual message in field 6. From now on we will call field 6 as AIS payload.

The data in AIS payload is an ASCII-encoded bit vector. Each character represents 6-bits of data from which can be recovered by subtracting 48 to its ASCII value. If the result is greater than 40, subtract again with 8 [23]. For instance, the 6-bits interpretation of the first 4 characters in the above AIS payload is `000001 000111 000111 011011`. By concatenating all 6-bits representation of each AIS payload character, we end up with binary payload of the data. The first 6 bits of the binary payload are the message type. There are 27 message types in total, but in practice the most common types emit by AIS transmitter are 1, 3, 4, 5, 18, and 24 [23]. Type 1, 3, 4, and 18 transmit position-related information while type 5 and 24 transmit static and voyage related information. In normal operation, an AIS transceiver broadcasts a new position-related message every 2 to 10 seconds while underway and every 3 minutes while stationary. Besides, a new static message is

sent every 6 minutes. The periodicity of the AIS message mentioned above does not happen in a real case. Most of the time the data is erroneous and irregular. We will discuss how to handle erroneous data later in the chapter.

Typically the amount of AIS data received from Singapore Straits contains 4 to 5 million records every month. The number could jump to 1 million records in a single busy day, but only 13,000 to 50,000 records on a quiet day. The last 10 days of the month are usually the busiest. Dynamic AIS message shares a common reporting structure for navigational information of total of 168 bits occupying AIS binary payload [23]. We will start by discussing every field in the AIS message. In general, every field belongs to either one of 3 categories, dynamic information, static information, and voyage-related information. The following list is part of dynamic information. Maritime Mobile Service Identity (MMSI) is a unique 9-digits identifier of AIS transceiver which usually stays in the vessel. Although MMSI is an AIS identifier, we could however use MMSI to identify a specific vessel. Navigational Status indicates vessel navigation activity around the sea. It explains whether a vessel is currently underway using engine, stationary at anchor, engaged in fishing, or other activities.

Rate-of-turn is values reported in degrees per minute between 0 to 708 which indicates the actual value of turning's rate of the vessel. Rate-of-turn in AIS message is encoded with a range of value from 0 to 128 whereby 0 means the ship is not turning whilst the plus-minus sign indicates the ship is turning right or left respectively. Given a rate-of-turn input (ROTin) from AIS message one can decode the actual ROT value by the following formula: ROT = 4.733 * SQRT(ROTin) degrees/min. Speed Over Ground (SOG) is the speed of the vessel relative to the surface of the earth. In the maritime world, there is another type of speed called Speed Over Water which is measured to water. The difference is speed-over-ground takes the speed of current into account which makes the total speed increase when the current flows in the same direction as the vessel and decreases when it flows astern, as opposed to speed-over-water that remains the same regardless of the current situation. AIS message's value for speed-over-ground is in 0.1 knot resolution from 0 to 102 knots, 1023 for not being available, and 1022 for speed over 102.2 knots.

Position accuracy is a 1-bit field in the AIS message to indicate if the vessel uses a DGPS sensor (value 1) or GNSS sensor (value 0, default) for the positioning

system. The position accuracy of DGPS is less than 10m while GNSS is more than 10m. To ensure that ship information being transmitted is correct and up to date, the shipowner shall validate the data regularly per month or voyage whichever shorter. However, the accuracy of the ship sensors into AIS would not be checked. The shipowner shall conduct a separate check during the voyage to validate the accuracy [12]. Longitude and Latitude is one of the most useful information in trajectory prediction. Longitude and Latitude's value in the AIS message is given in minute/10000 unit. The primary unit for longitude and latitude is in degree from -180 (west) to 180 (east) and -90 (west) to 90 (east) respectively, so we need to divide the original number with 600000 to convert them into a degree. Course Over Ground (COG) is the angle relative to true north ranging from 0 to 359 with 0.1 degree resolution. Heading or bearing is the magnetic compass angle indicated during voyage ranging from 0 to 359 degree value with value 511 means not available.

The second cluster of AIS fields is static information. Static and dynamic information share a few attributes in common such as timestamp, MMSI, and message type. IMO number, Callsign, and Ship Name primarily have the same function to uniquely identify the vessel. IMO number contains the 3 letters "IMO" followed by 7-digit-number which consists of 6 digits unique number and 1 last digit to verify the integrity of IMO number. This is done by multiplying each digit by a factor of 2 to 7 corresponding to their position from right to left. For instances, IMO 9074729: (9x7)+(0x6)+(7x5)+(4x4)+(7x3)+(2x2) = 139. In this case the IMO number is verified as both are the same (Vuori, 2013). A Callsign number is a 6-digit alphanumeric identity that is assigned by a national licensing authority to a vessel and acts in the same way as a registration number for a vehicle. The first 2 alphanumeric prefix of callsign number refers to nationality of the vessel and followed by 2 or 3 alphanumeric digits. The prefix for Singapore's licensed stations is '9V'. The last vessel identification system is the ship name. Each ship name may consist of multiple MMSI and IMO numbers, therefore it's not recommended to use ship name only for vessel identification purposes. Ship Type in AIS is encoded in 0 to 99. The most common ship type is cargo, tanker, passenger, fishing, and tug vessel. Another static information is the dimension of 4 sides of the ship including bow, stern, port and starboard measured in meters.

The last cluster in AIS fields is voyage-related information. Draught or Draft is

the distance from the waterline to the hull of the vessel. Draught value ensures a safe balance of the maximum load of the vessel to operate on a voyage. Estimation Time Arrival as the name suggests is an estimated number of vessels to arrive in a month, day, hour, and minute to a Destination. In practice, the information in Destination and ETA is not reliable, as it has to be manually updated by humans rather than gather automatically by sensors [23]. Such unreliable information is usually excluded from data processing for the prediction model. Now we have discussed all fields in the AIS message and hopefully will give us a better idea of what information we are trying to analyze.

3.2 Data Exploration and Statistics

We will explore AIS data to uncover more information as we progress this section. The whole process of discovering insight in data through statistics and visualization is called Exploratory Data Analysis (EDA). AIS data is essentially a collection of the trajectory of vessels across space and time, sometimes referred as spatio-temporal data.

Anita Graser *et al.* [7] describes a general workflow to dealing with trajectory data exploration in the following four steps.

1. *Establishing an overview* by visualizing raw input data records (including assessment of spatio-temporal extend and gaps in data)

2. *Putting records into context* by exploring information from consecutive movement data records with respect to time (e.g. speed, coordinates, or direction over time)

3. *Extracting trajectory, location, and events* by dividing the raw continuous tracks into individual trajectories, location, and events

4. *Exploring patterns and outliers* in trajectory and event data by looking at groups of trajectories or events.

Figure 3.1: Singapore Straits Map

	C0	type	mmsi	speed	accuracy	lon	lat	course	heading	time	...	day	hour	minute	draught	destination
0	0	18	563015550	0	0	103.728	1.30646	0	NA	2019-12-01T00:00:07Z	...	NA	NA	NA	NA	
1	1	1	563037440	0.2	0	103.762	1.29001	94.3	NA	2019-12-01T00:00:07Z	...	NA	NA	NA	NA	
2	2	18	563040772	12	0	103.759	1.29214	49.4	NA	2019-12-01T00:00:08Z	...	NA	NA	NA	NA	
3	3	1	566000010	0.2	1	103.761	1.2931	NA	120	2019-12-01T00:00:30Z	...	NA	NA	NA	NA	
4	4	1	255805665	16.8	0	103.757	1.13256	46.1	48	2019-12-01T00:00:30Z	...	NA	NA	NA	NA	

5 rows × 26 columns

Figure 3.2: AIS Raw Data

Table 3.1: AIS Data Quantity over Class Type

Class	Position Record	Static Record	Number of Ships
A	4,151,563	203,987	26,248
B	336,202	9,058	3,443
Total	4,487.765	213,045	29,691

3.2.1 Statistical Information

Our data was collected from NUS Maritime Data Center of School of Computing in Singapore ranging from August 2019 to December 2019. In this project, we limit our observation or area of interest to Singapore Straits as shown in Figure 3.1. The 1-month data consists of 4 million AIS records. Figure 3.2 shows a snapshot of the first 5 raw data, note many fields that come with null values as we will address the issue later on in the section. The quantity of AIS data by the class type is given in 3.1. In general AIS class type consist of Class A and Class B. Class A type is of type 1, 2, 3, and 5 while type 18 and 19 belong to Class B. Of all Class A type, type 1, 2, and 3 broadcast position report, and type 5 as we have discussed broadcast static report. Type 18 and 19 of Class B broadcast position report and static report respectively.

Table 3.2: Missing Values of Class A Position Report

Field	**Missing Values**	**Proportion**
MMSI	0	0.00%
Speed	28,788	0.70%
Accuracy	0	0.00%
Course	299,701	7.21%
Heading	1,330,678	32.05%
NavStatus	0	0.00%
ROT	0	0.00%
Longitude	0	0.00%
Latitude	0	0.00%

There are 25 fields currently used by our data center to store AIS records. Some of the fields are erroneous, missing values, and not entirely useful for trajectory analysis either because the fields are hand-updated by humans and hence susceptible to error, or the message sent by the transceiver is not accurate due to faulty sensor. The quality of AIS data is not only critical to the comprehensiveness of analysis but also an essential factor in avoiding misleading prediction results [30]. We will use several data pre-processing methods proposed by Zhao *et al* [30]. on AIS data and verify the result in the context of our area of interest. The summary of our findings is presented below.

First we will investigate missing values in AIS data. The missing value is defined as the data value that is not stored for a variable in the observation of interest [13]. To identify missing values, we observe the data per class type and compare it within the class. Table 3.2 and 3.3 summarise missing values by the field for each class of position reports. Class A static report (type 5) does not have a missing values, whilst the only field having missing value in a static report of class B (type 19) is Heading with 97% proportion. Considering missing values of all class yield the number as shown in Table 3.4. There are 3 fields that suffer from missing values including Speed, Course, and Heading in ascending order.

The further step of AIS data exploration is to identify erroneous value. One example of an obvious invalid value is longitude with more than 180 and latitude with more than 90. Following Harati-Mokhtari *et al.* [8] recommendation we evaluate the erroneous value of AIS records organized by the AIS field, the result is summarized in Table 3.5. Our findings suggest that erroneous value occupies a small percentage

Table 3.3: Missing Values of Class B Position Report

Field	Missing Values	Proportion
MMSI	0	0.00%
Speed	128	0.03%
Accuracy	0	0.00%
Course	32,871	9.77%
Heading	331,764	98.67%
Longitude	0	0.00%
Latitude	0	0.00%

Table 3.4: Missing Values Combined

Field	Class A	Proportion	Class B	Proportion
Speed	28,788	0.70%	128	0.30%
Course	299,701	7.21%	32,871	9.52%
Heading	1,330,678	32.05%	340,635	98.66%

Table 3.5: Erroneous Value by AIS Field

Field	Invalid Criteria	# Messages	Proportion (%)
MMSI	Digit's length < 9	1,002	3.45
ShipType	Undefined	587,375	14.14
CallSign	Non-alphanumeric prefix	2,750	73.41
NavStatus	Undefined	341,505	8.22
	Reserved for future use	80,974	1.95
Longitude	$<$ -180 or $>$ 180	21,660	0.48
Latitude	$<$ -90 or $>$ 90	21,660	0.48
Speed	$[102.3, \infty)$	0	0
Course	$[360, \infty)$	0	0
Heading	$[360, \infty)$	0	0
ROT	Undefined	1428,293	34.40
Draught	0 Length	12,919	6.33

of AIS data. We can mitigate the issue by emitting those values or designing the AIS module system that prevents users from entering invalid values.

3.2.2 Trajectory Data Exploration

Figure 3.3 is an overview of vessel trajectory over the first seven days of December in Singapore Straits. Trajectory coordinates is plotted using small dots with different color to indicate vessel type. The plot is showing some trajectory patterns, for instance the bottom trajectory is following 'v' letter and is dominated by cargo,

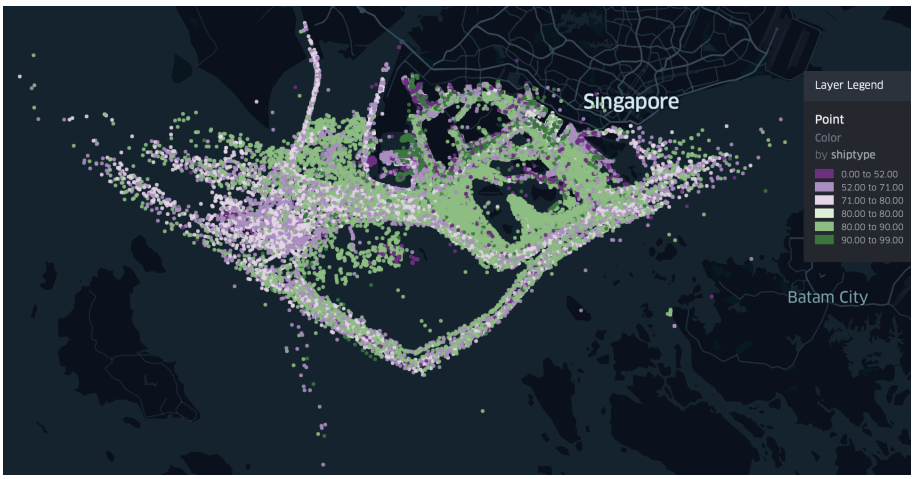


Figure 3.3: Vessel Coordinates, generated using kepler.gl

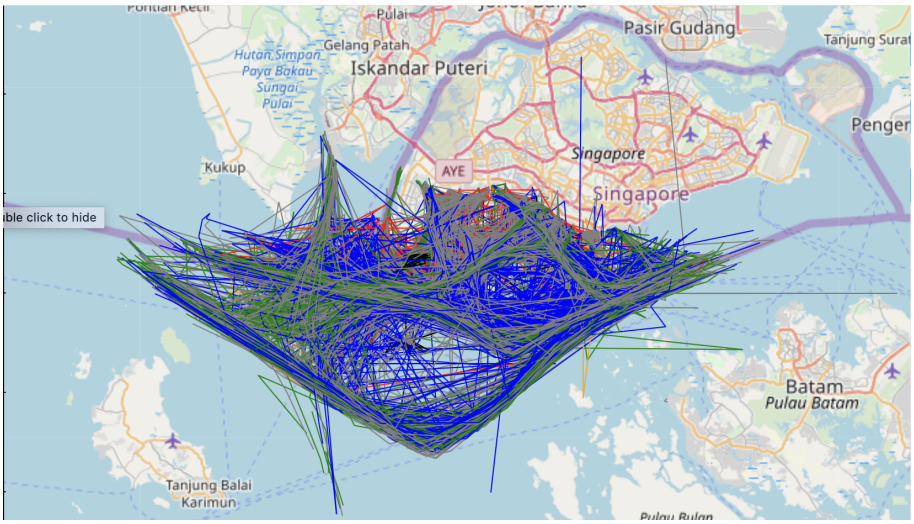


Figure 3.4: Coordinates, generated using movingpandas

tanker, and passengers vessels. It also shows that vessel follows and avoids certain route. A quick look on the trajectory of passengers vessel by using kepler.gl's filter tells us that two common routes being identified are Harbourfront Terminal to Batam Centre and Pasir Panjang Terminal to Bukom Island. We can infer a pattern that vessel from Batam to Harbourfront or the other way shall keep the distance close to Sentosa Island when about to dock or leave Harbourfront terminal.

Another way of visualizing vessel movement is to extract a collection of trajectories from datasets per MMSI. Each trajectory is represented by a curved continuous line with different color depending on the ship type. Figure 3.4 confirms similar trajectory pattern with Figure 3.3. The dominating color is green and blue which

represents Cargo and Tanker ships respectively.

AIS records is ideally broadcast by vessel every 2 to 10 second for class A dynamic type message, or every 6 minutes for static type. The AIS dataset we use for exploratory data analysis include dynamic attributes and ship type from static records that is joined by common MMSI. We found that 75% of dataset have time interval less than 16 minute, while the full interval distribution vary from as close as 0.96 second to as sparse as 6 day. We investigate if the given speed data is rational based on law of psychics. We do this by comparing the original speed with the derived version of speed, computed as distance per time, and then measure the difference. We found that the two version of speed is very close. The number suggest that 75% of the data differ by less than 0.91. Given the validity of the AIS speed data, we found that about 0.04% of container ships travel above the average speed of 24 knots, and the maximum recorded speed is 38.6 knots.

3.3 Summary

In this chapter, we discussed about AIS data exploration. All AIS features were demystified, from the definition, decoding the message, to the potential criteria of wrong value. We also explore the dataset using statistics and visualization. First categorize AIS records into two class of A and B. Then we calculate the number of records and unique ships in each class. We also investigate the number of message in all fields with missing and erroneous value according to certain criteria. Finally we conclude the chapter with trajectory exploration by visualizing the first seven week of data.

Chapter 4

AIS-based Trajectory Prediction Model

4.1 Preliminaries

In the data science pipeline, the next stage after data exploration and visualization is data preparation and transformation [6]. This chapter attempts to predict the future movements of the vessel using a prediction model. We will first prepare AIS datasets in a usable form and then proceed with the development of the model. In dealing with movement data, [16] and [7] provides a description of the process and tools that we use in the following sub-section.

4.1.1 Defining Ship Track

Recalled from the Data Description part in Chapter 3, AIS data consist of dynamic and static records. Dynamic records include a timestamp, MMSI, longitude, latitude, and other information. We use a data object called POINT implemented by Shapely library to transform longitude and latitude into a single coordinate object for easier object manipulation. We will use the term coordinate and POINT interchangeably for the rest of this chapter, and by that we are referring to longitude and latitude as well. The ship track is defined as a consecutive POINTS of the ship of various lengths. The behavior of tracks vary from one another; one track might be a continuous sailing within a short period, another might include months of sailing with multiple anchoring points [16]. We observe that many tracks consist of few POINTS. Shorter tracks less than 100 meters or 30 POINTS are excluded from the list to ensure our tracks carry enough information for the neural network to learn

the pattern. A track also would be split into two in the event of the time interval between consecutive records is more than 15 minutes. The 15 minutes tress-hold is derived from our finding that time interval between AIS consecutive records makes up from 0.96 seconds up to 6 days with 75% of the records account for 16 minutes of the time interval. The process of collecting, filtering, and splitting tracks from the unstructured AIS dataset is done using "movingpandas" library. Movingpandas [7] and Shapely are dedicated python package designed for spatial data analysis and manipulation.

4.1.2 Handling Missing and Erroneous Data

We summarised missing and erroneous value by the AIS field in the previous chapter and recalled that while longitude, latitude and MMSI do not have missing value, they both suffer from erroneous value by 3% and 5% respectively. In dealing with erroneous data, we filtered out those records with invalid value criteria as mentioned in Table 3.5 of Chapter 3. We also limit our data observation into the Singapore Straits region as an additional way to filter out the erroneous coordinates.

Spline and Linear interpolation are the two techniques we used to fill missing data. Interpolation is as an approach of constructing new variable values given a discrete set of its intermediate values. We use interpolation in response to the fact that the AIS time series is not evenly sampled and hence the data is considered missing at a certain point of time. According to [1] in [15], a useful interpolation technique should meet four criteria: (i) not a lot of data required to fill the missing values; (ii) estimation of parameters of the model are permitted; (iii) efficient data computation; (iv) the technique should applicable to stationary and non-stationary.

The spline is one of the interpolation technique with smoothing parameters. Smoothing is useful when there are many outliers in a dataset like AIS records so that the interpolated curve would focus on finding the main values. Outliers are a problem for the machine learning model especially if the numbers are high as the model would learn from false information and would output false inference. Linear interpolation is another interpolation that tries to fit a set of values using linear polynomial to construct new data. Spline and Linear interpolation are supported by Scipy, a python-based ecosystem for scientific computation. There are several Spline

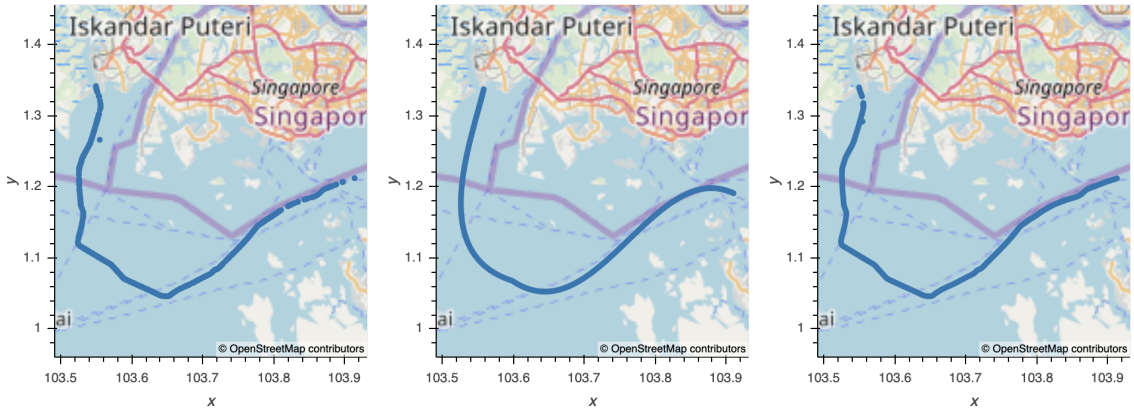


Figure 4.1: Trajectory interpolation of 2 methods. Left: Normal. Center: Spline. Right: Linear. Notice the single noise in the Left and Right

and Linear methods in Scipy depending on the problem, we use LSQ Univariate Spline for its one-dimensional spline interpolation with explicit internal *knots* and Interp1D for Linear interpolation. We observe that the Linear technique is less sensitive towards the outlier coordinate as compared to Spline. However, the Linear technique does not overly smooth-off the curve as the Spline does which is somewhat closer to the actual curve. Figure 4.1 shows the 3 trajectories as a comparison.

4.2 Methodology

The research question we need to answer in this project is, how can we develop a model to best predict the future movement of the vessel given their historic navigation information. To answer this question, we use the AIS time series data collected in Singapore Straits during December 2019 period. There are two most common approach for predicting time series models that can be applied to our problem; statistical and machine learning (including artificial neural network) approach. A statistical approach for time series prediction has been used in application such as stock market forecasting even before machine learning become popular. The objective of machine learning is the same as that of statistical ones. They both aim at improving prediction accuracy by minimizing some loss function, typically *mean absolute error* or *mean square error* [17]. The difference between the two lies in the fact that the statistical method assumes the data to be stationary and the algorithm is linear while the machine learning model assumes none of them. A stationary time

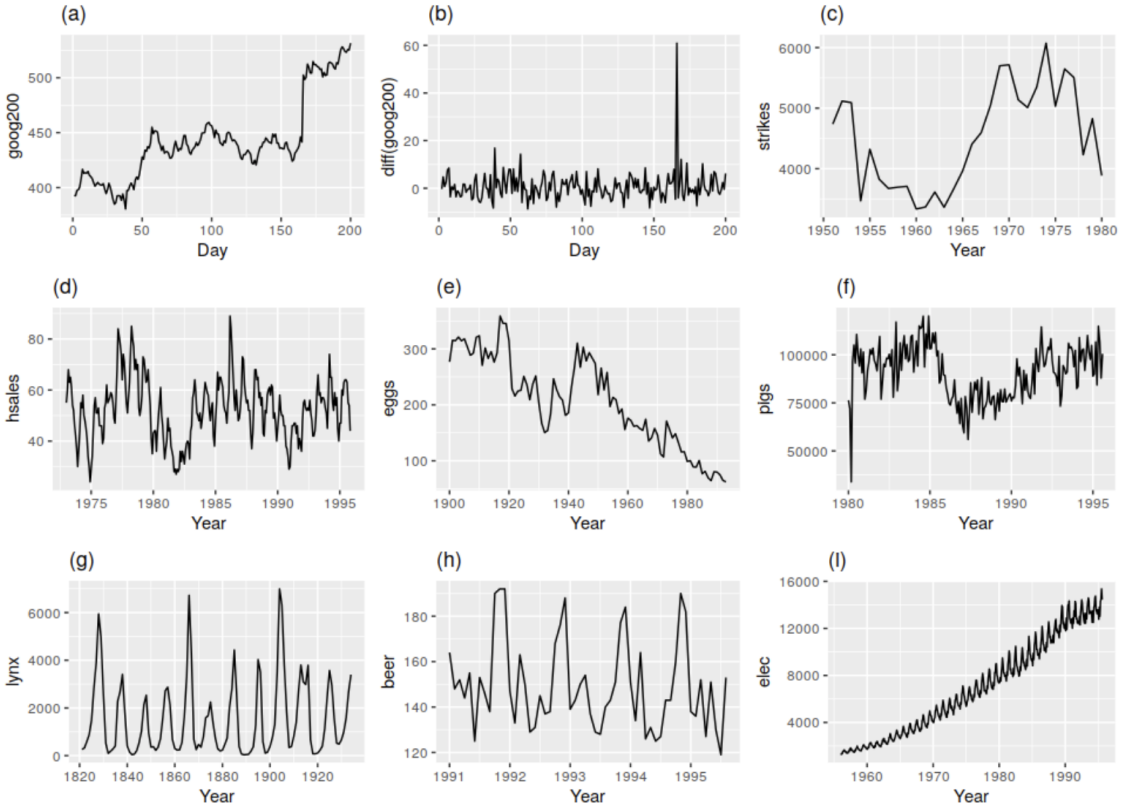


Figure 4.2: Which of these series are stationary? (a) Google stock price for 200 consecutive days; (b) Daily change in the Google stock price for 200 consecutive days; (c) Annual number of strikes in the US; (d) Monthly sales of new one-family houses sold in the US; (e) Annual price of a dozen eggs in the US (constant dollars); (f) Monthly total of pigs slaughtered in Victoria, Australia; (g) Annual total of lynx trapped in the McKenzie River district of north-west Canada; (h) Monthly Australian beer production; (i) Monthly Australian electricity production. (Figure taken from Forecasting: Principles and Practice by Rob J Hyndman)

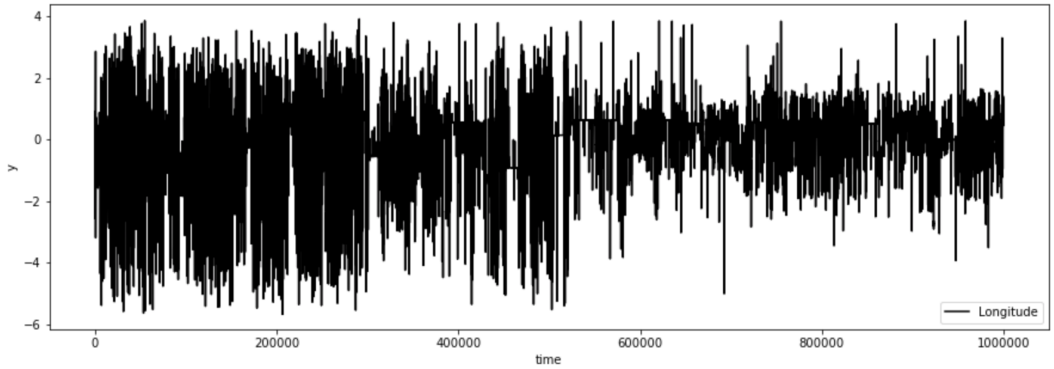


Figure 4.3: Longitude AIS time series data for 1 month

series is one whose statistical properties such as mean, variance, auto-correlation do not change over time. Thus, time series with trend and seasonality are not stationary [10]. Figure 4.2 shows some examples of stationary data over a different dataset. Figure 4.2.b and 4.2.g are the only stationery series while the rest are not.

Although a study by Makridakis *et al.* suggest that some statistical methods outperform machine learning and neural network model on M3 competition dataset, we decide to use machine learning models in the experiment of this chapter one reason; AIS time series data as shown in Figure 4.3 is not stationary. The data consist of many vessel types (e.g. passenger, cargo, tanker, etc) with different navigation status across a different period of voyage. That itself ruled out the data to be stationary. We will experiment using a simple machine learning model for baseline; a Linear Regression. The next model to develop an attempt to improve the accuracy of this baseline result.

With the machine learning model, the problem of predicting future vessel trajectory can be achieved with either regression (predicting value) or classification task (predicting class), depending on the output representation. The classification approach for movement data typically starts with trajectory clustering and proceeds with assigning a new trajectory to the closest cluster to decide where the next vessel's position is. The regression approach, on the other hand, predicts the actual value of the vessel's next position by using a set of input features and target labels. We believe that the classification approach is worth exploring but due to our time limitation we will focus on using the regression task for this thesis.

As we have narrowed down to use machine learning and neural network model, it is important to understand that the two models accept different input representations. Input representation is the way we transform the dataset into a usable form that is understandable by the model. The earlier section of 4 has pre-processed AIS data into a collection of tracks grouped by track id. The pre-processed dataset has 2584 unique MMSI and 10812 tracks over the course of 1 month AIS data. As seen in figure 4.4, the time series data consist of a two-time column, one in datetime format, another in an integer format. It also consists of 2 IDs to identify a coordinate, one is MMSI ID, another is Track ID where a single MMSI ID may consist of some Track ID as happened in a long voyage of the ship with the large time interval gaps being split into multiple tracks. Lastly the data is sorted by track id in increasing order.

time	timestamp	mmsi_id	track_id	longitude	latitude
2019-12-22 05:33:35	1576964015	567413000	27037	103.66920	1.19593
2019-12-21 01:35:04	1576863304	563032960	12556	103.74865	1.28898
2019-12-20 05:23:59	1576790639	566751000	26485	103.65561	1.24696
2019-12-30 01:15:54	1577639754	319017500	1817	103.66513	1.23283
2019-12-21 18:56:57	1576925817	566135000	24436	103.72225	1.21621

(a) Before sorted by track id

time	timestamp	mmsi_id	track_id	longitude	latitude
2019-12-04 21:23:20	1575465800	5631127	6	103.77934	1.28308
2019-12-04 21:35:29	1575466529	5631127	6	103.77925	1.28326
2019-12-04 21:37:30	1575466650	5631127	6	103.77931	1.28309
2019-12-04 21:39:03	1575466743	5631127	6	103.77925	1.28316
2019-12-04 21:45:43	1575467143	5631127	6	103.77936	1.28312

(b) After sorted by track id

Figure 4.4: Pre-preocessed AIS dataset

This will become our base dataset for the machine learning model.

Time series data has a property called sequence for their input in addition to samples and features. The sequence is a consecutive observation of input from the time step t backward. We define the number of observation sequences as the window size. Time series data also has a sequence of prediction from the time step t forwards. We define the number of prediction sequence as response size. For response size of 1, we called it a single-step prediction and for response size of more than 1, we called it a multi-step prediction. It is also possible for time series data to have multiple variables prediction instead of just one, hence called multivariable prediction. The problem we want to solve is characterized as a multi-step and multivariable prediction. If we look at AIS data in Figure 4.4, our time series data is not evenly sampled or irregular. There are two ways to handle the irregularity of time series, first is to downsample or oversample the data with a specified period (e.g., minute, hour, day) using interpolation as we discussed in Section 4.1.3. Downsampling data would result in generating more data while oversampling data on the contrary is reducing the number of data. Secondly, it is to keep the irregular time series unaltered and let the prediction model learn the time dependency. This can be achieved by inserting a time difference between timestamp t+1 and t as a new input feature in addition to longitude and latitude.

4.2.1 Machine Learning Input Representation

Machine Learning model accepts 2-dimensional input and 1-dimensional output. The input is the number of samples of several features while the output is the number of samples of one target variable. The two-dimension input and one dimension output are often called supervised learning problems. It is not the nature of the machine learning model to work with time-series data that involves window size and response

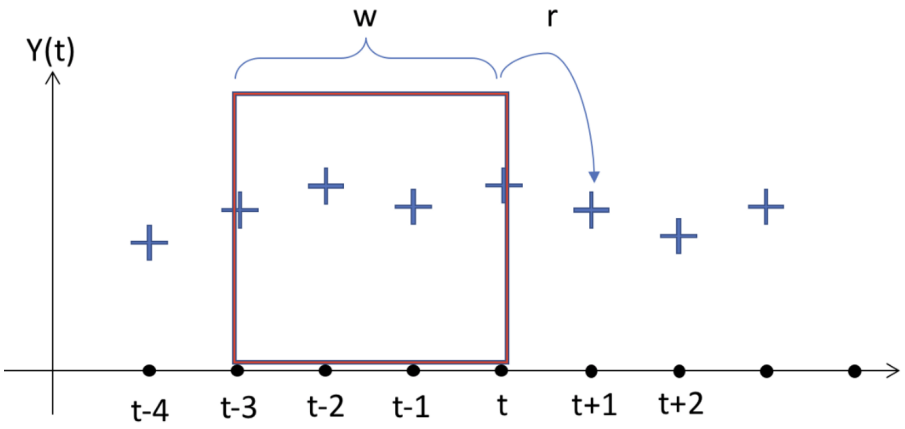


Figure 4.5: Sliding window method to transform time series into sequential supervised learning input. Figure taken from Pablo Ruiz

size of more than 1. In the need of solving the time series problem with the machine learning model, we need to phrase it as a sequential supervised learning problem by using the sliding window method. The *sequential supervised learning* problem can be formulated as follows. Let $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N}$ be a set of N training examples. Each example is a pair of sequence $(\mathbf{x}_i, \mathbf{y}_i)$, where $\mathbf{x}_i = \langle \mathbf{x}_{i,1}, \mathbf{x}_{i,2}, \mathbf{x}_{i,3}, ..., \mathbf{x}_{i,T_w} \rangle$ and $\mathbf{y}_i = \langle \mathbf{y}_{i,1}, \mathbf{y}_{i,2}, \mathbf{y}_{i,3}, ..., \mathbf{y}_{i,T_r} \rangle$ where w and r represents windows size and response size. The goal is to construct a classifier h that can correctly predict a new label sequence $\mathbf{y} = h(\mathbf{x})$ given an input sequence of x [3]. The sliding window method is illustrated in Figure 4.5 where the window size is 4 and the response size is 1. We transform each sequence of time series data from the 3-dimensional shape into 2 dimensional one. For the first sequence, the sliding window observes features from time step t to t-3 and indicates the target variable at t+1 as the output. The result is a metric of shape 4×3. Next the metrics need to be flattened into a shape of 1×12 allowing each observation at any time-step t become a feature of its own. The sliding process continues from time-step t+1 to t-2, t+2 to t-1 and so on until all dataset has been covered. Our final dataset transformation is presented in Figure 4.6b with the original dataset in Figure 4.6a. The blue box and red box in Figure 4.6a represent a set of input and output for a single observation. Each column values are flattened and become new features in Figure 4.6b. The last input feature delta-t(5) is added to indicate that the output is taken from time-step 5 and hopefully the model could learn the pattern. The output Y in Figure 4.6b comes from the red

	Δt	lon	lat
0	-0.539013	0.896525	0.673833
1	7.039163	0.895196	0.677396
2	0.718819	0.896082	0.674031
3	0.427750	0.895196	0.675416
4	3.619108	0.896820	0.674625
5	5.760541	0.896082	0.673635
6	3.307249	0.897706	0.675416
7	1.228189	0.895639	0.674823
8	1.955860	0.896377	0.674823
9	0.708423	0.895196	0.674031

(a) The original dataset before feature transformation has 3 features

	Δt(1)	Δt(2)	Δt(3)	Δt(4)	lon(1)	lon(2)	lon(3)	lon(4)	lat(1)	lat(2)	lat(3)	lat(4)	Δt(5)	Y
0	0.0	7.039163	7.757982	8.185732	0.896525	0.895196	0.896082	0.895196	0.673833	0.677396	0.674031	0.675416	11.804840	0.896820
1	0.0	0.718819	1.146569	4.765677	0.895196	0.896082	0.895196	0.896820	0.677396	0.674031	0.675416	0.674625	10.526218	0.896082
2	0.0	0.427750	4.046859	9.807399	0.896082	0.895196	0.896820	0.896082	0.674031	0.675416	0.674625	0.673635	13.114649	0.897706
3	0.0	3.619108	9.379649	12.686898	0.895196	0.896820	0.896082	0.897706	0.675416	0.674625	0.673635	0.675416	13.915087	0.895639
4	0.0	5.760541	9.067790	10.295979	0.896820	0.896082	0.897706	0.895639	0.674625	0.673635	0.675416	0.674823	12.251838	0.896377

(b) The transformed dataset has total 13 input features and 1 output target out of window size 4 and response size 1

Figure 4.6: AIS dataset transformation

box of either longitude or latitude feature.

4.2.2 Neural Network Input Representation

Our neural network model consists of a stack of Long Short Term Memory (LSTM) and Dense layer. The input-output of the LSTM and Dense layer differ slightly as compared to the machine learning one. To work with our mentioned model, we need to transform the input and target dataset into a sequence of coordinate using a defined window size N and response size R. This can be achieved by shaping the input dataset in 3-dimension where the first dimension represents the batch size, the second dimension represents the window size N from time step t to t-N-1, and the third dimension represents the number of feature in the input sequence, which in our case 2 features for longitude and latitude. The target dataset would have a shape of 2-dimension where the first dimension represents the response size R from

timestep t+1 to t+R, and the second dimension represents the number of features in the target sequence, which also happens to be longitude and latitude. The process of creating an input sequence is illustrated in Figure 4.6a using window size N of 4 and response size R of 1. The total new input size is M-N-R+1 given the number of datasets is M.

4.2.3 Loss Function and Evaluation Metric

Loss Function and Evaluation Metric often share the same definition; they both are a method of evaluating model performance quantitatively, but they serve a different purpose. The loss function is used to optimize the model during training while the evaluation metric is used to measure the performance of the model. The former is a reference for an optimization algorithm to help reduce the error, while the other one has no relation to the optimization algorithm. In the regression problem, the function used for loss function can also be used as an evaluation metric. The most commonly used loss function for regression are *Mean Absolute Error* and *Root Mean Squared Error*, and we add more evaluation metric called *Mean Absolute Percentage Error*.

- Mean Absolute Error (MAE) or often called L1 loss, is measured as the average sum of the absolute difference between prediction and the actual value. Since MAE take the absolute difference as the loss, it is considered robust towards outlier and hence suitable for a dataset with many outlier or noise. Mathematically calculated using the following formula: $\text{MAE} = \frac{1}{n}\Sigma_{i=1}^{n}|y_i - \hat{y}_i|$.

- Root Mean Squared Error (RMSE), is measured as the square root of the average sum of the squared difference between prediction and the actual value. If the difference between prediction and the actual value is large, perhaps due to the outlier, the root square of it would be larger as compared to the absolute error approach. Therefore RMSE is preferred if we want to penalize large errors. This would result in being wrong on outlier input prediction. Mathematically calculated using the following formula: $\text{RMSE} = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}$.

- Mean Absolute Percentage Error (MAPE), is the percentage equivalent of Mean Absolute Error which is more intuitive to interpret. In general, MAPE

of more than 50% is considered inaccurate forecasting, between 20% to 50% is reasonable, between 10% to 20% is good, and less than 10% is high accuracy forecasting. The mathematical formula is given: MAPE $= \frac{1}{n}\Sigma_{i=1}^{n}|\frac{y_i - \hat{y}_i}{y_i}| \times 100\%$.

AIS time series data consist of an outlier due to erroneous value in longitude and latitude. Because of this, we use MAE as the loss function of our model and use MAE, RMSE, and MAPE as theevaluation of the metrics to measure the performance of our model.

4.2.4 Linear Regression Model

Thad Starner, a Computing Professor at Georgia Tech, once said "Always do the simple thing first. Only apply intelligence when required", which emphasizes the need of solving a problem with the simplest solution first.

A baseline model is simple and easy to set up and yet has the potential to deliver decent performance. In machine learning, the baseline model is used as a starting reference from which we decide to develop a better model with improved performance. Since the definition of the baseline is broad, a different problem would need different baseline depending on what data we have and what task we want to solve.

Linear Regression is among the simplest model to test out baseline performance. Linear Regression model the relationship between an observed variable and its dependence variables. To use Linear Regression, the AIS dataset has been transformed following the discussion in 4.2.1 about machine learning input representation. We choose to use time, longitude, and latitude as the features and the target variable is either longitude or latitude. It is chosen as consequences of the machine learning model, including Linear Regression, which is limited to predict only a single variable. In this case we create 2 models where one model is for predicting longitude while another is for predicting latitude.

Since we are interested in predicting longitude and latitude, we train the model and evaluate the error independently for each variable. The final score is the sum of each error divided by total length.

We experiment with some observation and prediction time steps. The result of the baseline performance for single-step prediction is presented in Table 4.1. Starting

Table 4.1: Baseline Model Performance for Single Step Prediction

Window Size	**MAE**	**RMSE**	**MAPE (%)**
30	0.0131	0.0732	9.73
20	0.0135	0.0741	9.87
10	0.0141	0.0762	10.15
1	0.0127	0.0901	7.50

Table 4.2: Baseline Model Performance for Multi Step Prediction

Response Size	**MAE**	**RMSE**	**MAPE (%)**
30	0.0978	0.2150	76.54
20	0.0705	0.1690	58.27
10	0.0411	0.1182	30.11

from window size of 1, all error metrics result in high accuracy with 0.0127 of MAE, 0.0901 of RMSE, and 7% of MAPE. The next window size of 10 does not improve the accuracy according to MAE metrics as the score increase to 0.0141. However the RMSE score decrease to 0.0762 as an indication of performance improvement. Now as the window size increase, all error metrics decrease. We also experiment with multi-step prediction, namely to predict 10, 20 and 30 steps forward with a window size of 30. The results as shown in Table 4.2 suggest that predicting 10 steps forward results in 0.0411 of MAE, 0.1182 of RMSE and brings the error accuracy up to 30%. Predicting 20 and 30 steps ahead would further severe the performance as the MAE and RMSE increase to 0.0978 and 0.2150 with error accuracy soar to 76%.

The Linear Regression model was very fast to train from 0.6 seconds to 90 seconds. These results will be the minimum performance for the next model to aim as we develop a more complex model in the subsequent section.

4.2.5 Recurrent Neural Network Model

The architecture of our model consists of several layers and we are going to follow [16] architecture recommendation with some modification. We use one of the RNN archetypes called Long Short Term Memory (LSTM) as the main layer because LSTM is designed to remember the context of the long sequence due to an additional memory gate in the network. The Dense layer is used as an output layer with Rectified Linear Unit (ReLU) activation function, allowing for N output vector prediction, where N is the number of feature prediction *times* the number of

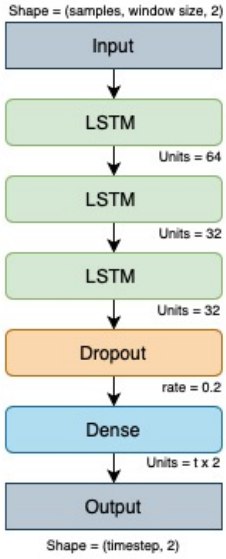
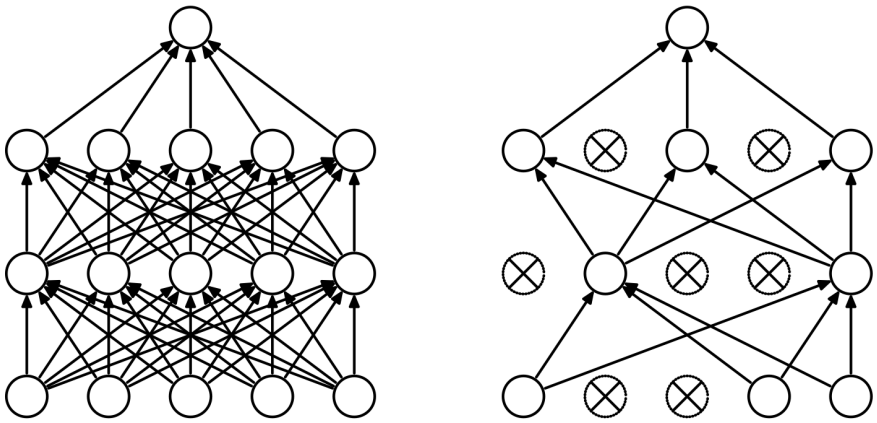


Figure 4.7: Illustration of the proposed model

prediction step. We may need to use the Dropout layer before the Dense layer if overfitting may happen. The full model architecture is shown in Figure 4.7.

Dropout is one of the most effective and commonly used regularization techniques to prevent overfitting from large networks. Overfitting is a condition where the model performs well on the training set but poor on the testing set. The term "dropout" refers to dropping out units and the connection in a neural network temporarily as shown in Figure 4.2 [27]. By temporarily means the dropped-out units would be returned along with the incoming and outgoing connections after the training phase has finished. In Keras, we can introduce Dropout in a network via the tf.keras.layer.Dropout API, which is applied to the output of the layer right before it [2]. *Dropout rate* value of 20% is often used to compromise between retaining model accuracy and preventing overfitting.

Every layer mentioned above except for Dropout comes with an activation function. Activation functions are function that takes in an argument of input, weight, and overall bias of neuron and decides a specific output value or activation value of the node. Activation function can be either linear or non-linear function depending on the function it represents [19]. The linear activation function is amongst the simplest yet limited in functionality due to its linear transformation. Non-linear activation is used to learn more complex functions. In general, the two most widely used non-linear activation are sigmoid and hyperbolic tangent



(a) Standard Neural Net (b) After applying dropout.

Figure 4.8: Dropout Unit Model taken from Srivastava *et al.* Left: A standard neural network of 2 layers. Right: Neural network after some units has been dropped along with its connection.

functions. The input to the sigmoid function is transformed into a value between 0 and 1 following an S shape-like curve, which means any value larger than 1 is equal to 1, and a negative value lesser than 0 become 0. Hyperbolic tangent has similar properties as sigmoid with an output value range between -1 to 1. In Keras, default activation function of LSTM is hyperbolic tangent (or *tanh*) and no default function for Dense layer. Generally, we use Rectified Linear Unit (ReLU) function for regression problems for 2 reasons, first ReLU has linear function property for input > 0 which is useful to avoid vanishing gradient problem, and secondly ReLU is a non-linear function.

4.2.6 Optimization Algorithm

The optimization algorithm helps us minimize or maximize loss function (sometimes referred to as the objective function) of the model. There are two major types of optimization algorithms; first-order and second-order optimization. The first-order optimization minimizes the function using *gradient* concerning its function parameters, while the second-order optimization uses *second order derivatives* to find the global minimum of its function. Our model applies a first-order optimization technique as it is more commonly used in deep learning practice. Gradient Descent is the name of the first-order optimization algorithm for finding local minimum by computing the first order derivative of loss function or often referred to as the gradi-

ent. The drawbacks of Gradient Descent are mainly being slow in convergence and the algorithm could be stuck in a sub-optimal local minimum. There are 3 variants of Gradient Descent to help resolve its problem; Batch Gradient Descent, Stochastic Gradient Descent (SGD), and Mini-batch Gradient Descent. The difference of each variant mainly in the amount of data being used to compute the gradient of loss function which results in the improvement of computing efficiency and accuracy. However it does not guarantee the algorithm to converge and find the local minimum. Several algorithms have been developed to address such problems by making use of gradient acceleration; including Momentum, Adagrad, Adadelta, RMSProp, Adam, and Nesterov Gradient. Andrej Karpathy, Director of AI at Tesla, analyzed the trend of optimization algorithms used in Arxiv papers from the course of 2012 until 2017 and found out that Adam is used in about 23% of papers, the highest among other algorithms. RMSProp comes second at 5%, followed by Adagrad and Adadelta at 4%. The popularity of Adam and RMSProp in neural network research happens for a reason. We discuss several algorithms related to the development of Adam and RMSProp.

- Momentum [22] is an algorithm that helps accelerate the convergence of SGD in the relevant direction using history movement and dampens the oscillation when the gradient changes direction.
- Adaptive Gradient or Adagrad [5] is an algorithm with an adaptive learning rates which means it computes individual learning rate for different parameters. The learning rate is low for parameters associated with frequently occurring features and set high for a parameter associated with infrequently occurring features [24].
- RMSProp is an adaptive learning rate method proposed by Geoffrey Hinton to solve the diminishing learning rate problem in Adagrad.
- Adaptive Moment Estimation or Adam [14] can be seen as a combination of Momentum and RMSProp technique resulting in faster convergence with all benefit from RMSProp and Adagrad.

Looking at the above discussion, our model will use Adam for the first attempt and switch to RMSProp in the event of a poor result.

4.3 Summary

In this chapter, we gave an overview of the research problem, the methodology we use, and the rationale behind each design choice. An input of the AIS time series dataset was built for two different models of representation, which can be applied to machine learning and recurrent neural network model. We design and implement a baseline model for a problem in hand, which allows us to build a better model in the next chapter.

Chapter 5

Result and Evaluation

5.1 Preliminary

In this chapter, experiments are conducted and performances are evaluated upon the proposed model. We analyze and compare the result of each experiment using a pre-defined metric evaluation. The setup of the experiment is presented in Section 5.2. The results of experiments are presented in Section 5.3, 5.4, and 5.5 where 3 experiments are conducted. Lastly in Section 5.6 we compare 2 different ways of generating a prediction of the selected model.

5.2 Experiment

We investigate different factors that potentially affect the performance of the prediction model, including several window size and response size. We also analyze the evaluation metrics using two interpolation techniques; Spline and Linear interpolation. All development and testing stage are performed via Jupyter Notebook and PyCharm IDE on MacBook-Pro 2.9GHz Dual-Core Intel Core i5, 8GB of RAM, and Intel Iris Graphics 550, while the model training of 1-month AIS dataset is run on Desktop PC with AMD Ryzen 7 1700 3.0 GHz Eight-Core AM4 processor, 64GB of RAM and GeForce GTX 1080 Ti GPU. We use the following libraries and framework in Python during the course of development; Tensorflow 2.0 for training LSTM model, Scikit-learn for data pre-processing and linear regression modeling, Geopandas for transforming longitude and latitude value into a single coordinate object, Movingpandas for the trajectory analysis and visualization, Kepler.gl for geospatial

analytic visualization on the web application, Scipy for variable interpolation, and finally Pandas for data analysis and manipulation. Our final datasets are split into 60% of training, 20% of validation, and 20% of the testing set which translates into 1505106, 501702, and 501702 total datasets respectively. The input-output pair of datasets are generated using a slice generator of tensorflow with shuffling.

We experiment with single-step and multi-step prediction time given a variety of window sizes. Multi-step prediction includes 10 and 30 timesteps in the future. There are two strategies to perform multi-step prediction in AIS time series dataset; (i) direct multi-step and (ii) recursive multi-step. The first part of the experiment uses direct multi-step strategy, and the second part uses recursive multi-step. Direct multi-step involves several models to predict each time step, which means to predict 10 time-step into the future we need to create 10 models; one model for each prediction step. Recursive multi-step in contrary involves only one model being used multiple times where the prior timestep is used as the input for the next prediction, which means to predict 30 timesteps we can either use 10 timestep model for 3 times or single timestep model for 30 times. During the experiment, a model training might stop in a different number of epochs depending on whether the error has reached the limit of early stopping criteria, which is set to 0.0001 deltas for 5 consecutive epochs. Early stopping is a technique introduced in tensorflow to prevent overfitting by stopping the training process when the loss does not improve for multiple stages of training or epochs.

5.3 Prediction Model Using 1 Timestep

We begin our experiment with 1 timestep prediction given several window size of ship movement from 1 to 30 timestep. Our input and target dataset for each sequence are shaped into $W \times 2$ and 1×2 where W represents a different value of window size and 1 represent the prediction timestep over 2 features. The input and target shape is applied to training, validation, and testing datasets.

Using Linear interpolation and a window size of 30, our model performance results in 0.0049 MAE and 0.0056 RMSE which translates to the lowest MAPE of 3.50% over 9 epochs. Lowering the window size to 20, 10 and 1 increase the error of the model as seen in Table 5.1. Compared to other window sizes, a window size of 1

Table 5.1: Model performance with dynamic window size and 1 prediction timestep

Window Size	MAE		RMSE		MAPE (%)		Epochs	
	Linear	Spline	Linear	Spline	Linear	Spline	Linear	Spline
30	0.0049	0.0049	0.0056	0.0057	3.50	3.42	9	12
20	0.0047	0.0053	0.0053	0.0059	3.57	7.34	18	10
10	0.0063	0.0043	0.0070	0.0048	4.37	3.71	9	13
1	0.0101	0.0115	0.0112	0.0128	8.53	12.32	7	10

suggests the highest error of 0.0101 MAE. The reason is that neural network does not have enough information to learn the vessel's movement pattern with just one value of observation. The experiment result with Spline interpolation is showing a different trend. Using a window size of 30 the error starts with 0.0049 MAE and 0.0057 RMSE, then go up slightly to 0.0053 MAE and 0.0059 RMSE on the window size of 20, decrease again to 0.0043 MAE and 0.0048 RMSE on the window size of 10, before another increase to 0.0115 MAE and 0.0128 RMSE on the window size of 1.

Figure 4.3 illustrate the plot between the observation (blue), prediction (green), and actual target (red) for 1 minute ahead. Although the error of predicting one timestep into the future is low, it might not be useful in practice. Therefore in the last experiment of section 5.6, we generate bigger prediction timestep recursively using single-step prediction model.

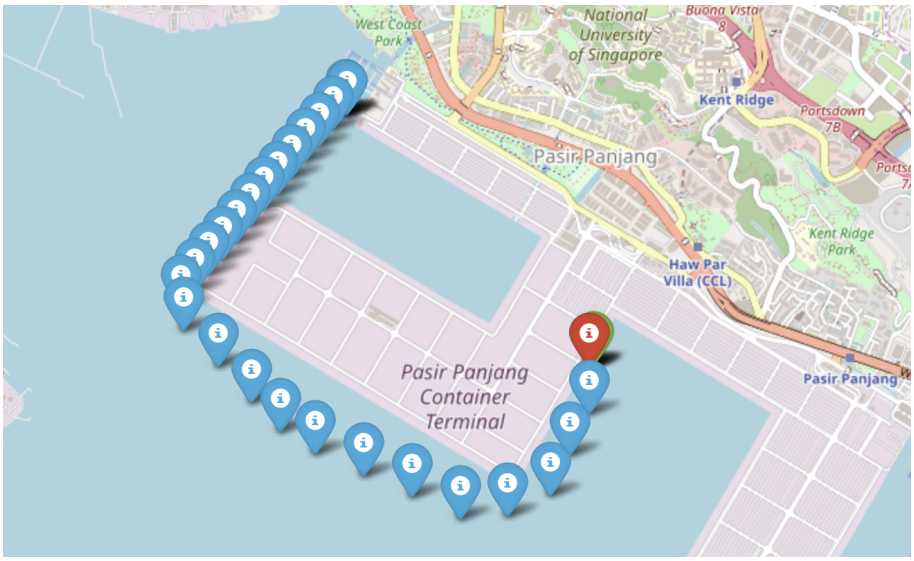


Figure 5.1: Plot of 1 timestep prediction given window size of 30

5.4 Prediction Model Using 10 Timestep

In this experiment, we develop a model with 10 prediction timestep given a variety of window size. Following the previous experiment, the shape of input and target is $W \times 2$ and 10×2 where W indicate the various value of window size and 10 indicate prediction timestep over 2 features of longitude and latitude. The input and target shape is applied to training, validation, and testing datasets.

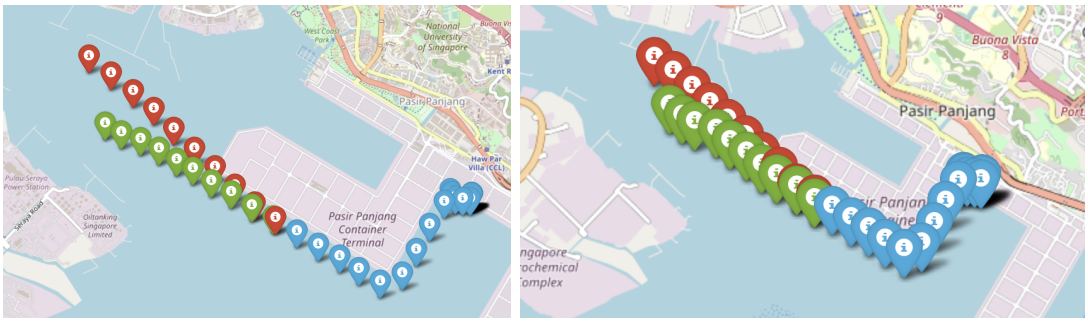
Using linear interpolation and a window size of 1, the error of the model is 0.0463 MAE, 0.0593 RMSE and 41% MAPE, the highest error among other window sizes. Once we increase the window size into 10 we see a significant drop in the error with 0.0233 MAE, 0.0332 RMSE and 19% MAPE. As we further increases the window size by 10 up to 40, the error slightly increase to 0.0238 MAE, 0.0339 RMSE and 20% MAPE as seen in Table 5.2. The result also suggests that increasing window size to 60 and 80 helps to decrease the error relatively small in MAE, RMSE, and MAPE as compared to the window size of 40.

With spline interpolation, the result suggests a different performance. Other than window size of 1, the MAE has an upward trend as the window size goes up until 80. Table 5.2 clarifies this point. It starts at 0.0169 MAE and 1.0240 RMSE on the window size of 10 and 20 and continuously goes up until 0.0195 MAE and 0.0269 RMSE on the last window size. The MAPE metrics on spline does not correlate proportionally with MAE despite MAPE is a percentage form of MAE. On window size of 10 and 20, while the MAE share the same error, the MAPE differ very significantly by 29% and 103% respectively. On window size of 30, the MAE increase from the previous window size while the MAPE however decrease. The highest MAE and the lowest MAPE come from the same window size of 80.

The trend of MAPE with spline interpolation might indicate that MAPE as a performance metric, in general, has a pitfall for some reasons. First, MAPE would fail if some of the actual values are zero. Second, MAPE would result in extremely high number if some of the actual values are very close to zero and therefore bias the informativeness of MAPE [11]. The issue could happen to any dataset, including AIS time series, being standardized using mean and standard deviation as the transformation would be centred in 0 points. We shall take the information from MAPE metrics with a grain of salt and use MAE and RMSE for cross-validation.

Table 5.2: Model performance with dynamic window size and 10 prediction timestep

Window Size	MAE		RMSE		MAPE (%)		Epochs	
	Linear	Spline	Linear	Spline	Linear	Spline	Linear	Spline
80	0.0225	0.0195	0.0320	0.0269	19.39	14.02	35	24
60	0.0229	0.0172	0.0326	0.0242	19.97	77.22	32	42
40	0.0238	0.0173	0.0339	0.0242	20.04	58.59	27	34
30	0.0239	0.0181	0.0340	0.0256	20.05	87.57	27	34
20	0.0234	0.0169	0.0333	0.0240	19.68	103.08	27	50
10	0.0232	0.0170	0.0332	0.0240	19.10	29.46	36	50
1	0.0463	0.0590	0.0593	0.0744	41.15	70.47	50	14



(a) Plot of 10 timestep prediction given window size of 30

(b) Plot of 10 timestep prediction given window size of 20

Figure 5.2: Plot of 10 timestep prediction

Predicting the vessel's position for the next 10 timesteps is certainly more useful in practice than just 1 timestep. To help us better visualize the model performance, we plot a certain ship track from the prediction model using linear interpolation. The blue, red, and green colour in Figure 5.2 represents observation, actual target, and prediction respectively. Figure 5.2 shows a trajectory with 10 timestep prediction and the window size of 30 and 20. Figure 5.2b shows more accurate prediction as compared to 5.2a.

5.5 Prediction Model Using 30 Timestep

Lastly, we experiment with 30 prediction timestep given a variety of window size. Our input and target dataset for each sequence are shaped into $W \times 2$ and 30×2 where W represents a different value of window size and 30 represent the prediction timestep over 2 features, longitude and latitude. This input and target shape is applied to training, validation, and testing datasets.

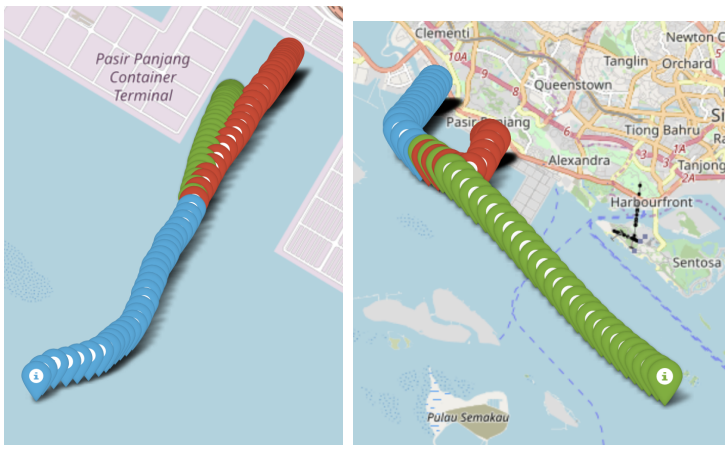
Table 5.3: Model performance with dynamic window size and 30 prediction timestep

Window Size	MAE		RMSE		MAPE (%)		Epochs	
	Linear	Spline	Linear	Spline	Linear	Spline	Linear	Spline
80	0.0697	0.0751	0.0981	0.1053	63.96	70.29	35	45
60	0.0746	0.0782	0.1041	0.1098	68.41	73.61	25	37
40	0.0744	0.0772	0.1044	0.1081	67.70	70.54	21	50
30	0.0762	0.0770	0.1070	0.1080	67.96	71.18	26	50
20	0.0772	0.0784	0.1082	0.1098	68.88	72.30	26	50
10	0.0840	0.0803	0.1182	0.1125	71.75	73.09	22	50
1	0.1426	0.1346	0.1790	0.1701	114.26	108.97	13	22

With linear interpolation, beside the window size of 1, the error of the model decrease from 0.0840 MAE, 0.1182 RMSE and 71.75% MAPE to 0.0697 MAE, 0.0981 RMSE, and 63.96% MAPE as we increase the window size from 10 to 80 as seen in Table 5.3. With spline interpolation, the performance metrics indicate the same trend as the one in linear interpolation. Beside window size of 1, the error starts at 0.0803 MAE, 0.1125 RMSE, and 73% MAPE on the window size of 10, and progressively decrease to 0.0751 MAE, 0.1053 RMSE, and 70.29% MAPE on window size 80. Looking at the MAPE, all window size in this experiment are categorized as an inaccurate model.

We can see from the plot in Figure 5.4a and Figure 5.4b, with window size 30, the predicted trajectory suffer from big deviation when the past trajectory tends to be non-straight. In contrary, the prediction is more accurate when the past trajectory is straight. We plot the same track on window size 40 which overall has

Figure 5.3: Plot of 30 timestep prediction using window size of 30



(a) Plot of 30 timestep prediction for straight past trajectory

(b) Plot of 30 timstep prediction for non-straight past trajectory

Figure 5.4: Plot of 30 timestep prediction with window size of 30

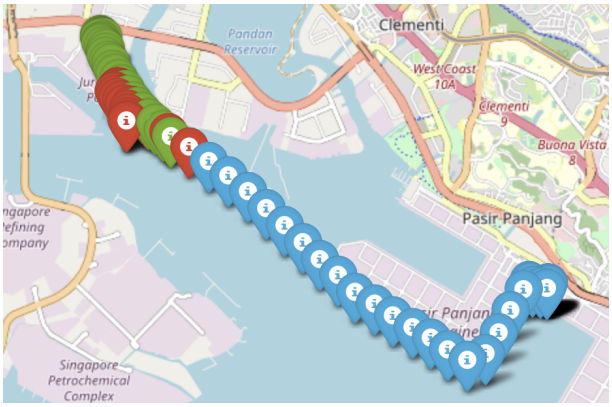
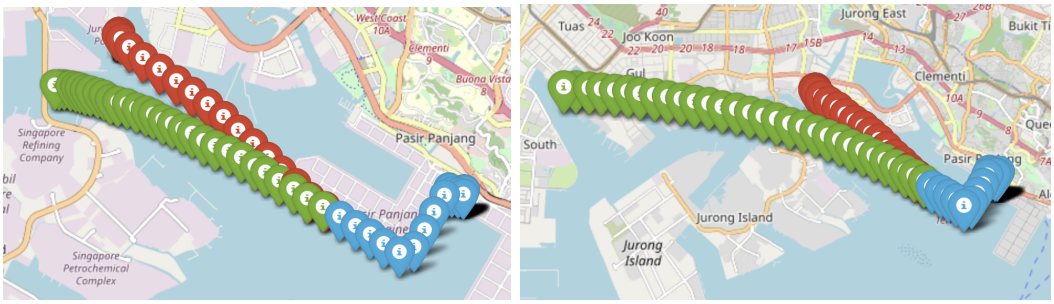


Figure 5.5: Plot of 30 timetep prediction using window size of 40

lower error than window size 30. Having longer window size help reduce the error as the predicted track tend to be closer to the actual track as seen in Figure 5.5.

5.6 Recursive Multi-step Prediction

Recall that recursive multi-step prediction makes use of the same model to generate multiple predictions recursively, meaning the output from the previous timestep become the input in the next prediction. In this experiment, we generate 30 timestep prediction with 3 methods, (i) direct prediction with 30 timesteps, (ii) recursive prediction with 10 timestep model for 3 times, (iii) recursive prediction with 1 timestep model for 30 times. The first method is the one we see in section



(a) Plot of 30 timestep prediction using 10 timestep model

(b) Plot of 30 timestep prediction using 1 timestep model

Figure 5.6: Plot of 30 timestep prediction using window size of 30

(a) Plot of 30 timestep prediction using 10 timestep model

(b) Plot of 30 timestep prediction using 1 timestep model

Figure 5.7: Plot of 30 timestep prediction using window size of 20

(a) Plot of 30 timestep prediction using 10 timestep model

(b) Plot of 30 timestep prediction using 1 timestep model

Figure 5.8: Plot 30 timestep prediction recursively using window size of 10

5.5 and Figure 5.3. To allow comparable result we pick one model in previous experiments from the same window size, evaluate the performance metrics and plot the track for visualization.

Using a window size of 30, the second method suggests a better performance

looking at the error scores; 0.0958 MAE, 0.1234 RMSE, and 24.24% MAPE. The third method did not improve the performance as the error score suggests at 0.2021 MAE, 0.3332 RMSE, and 151.64% MAPE The plot to validate this result is presented in Figure 5.6.

Using windows size of 20, the second method gives even lower error score compared to the same method on the window size of 30. Its performance metrics results in 0.0628 MAE, 0.0833, and 11.81% MAPE. The third method's performance metrics score relatively the same as the second method at 0.0619 MAE, 0.0820 RMSE, and 68.33% MAPE. Figure 5.7 helps us visualize the difference between the two methods.

In the last experiment with a window size of 10, the error in the second method score at 0.0877 MAE, 0.1142 RMSE, and 17.23% MAPE, slightly higher than the same method on the window size of 20. The performance metrics of the third method is significantly higher as compared to the first method at 0.3033 MAE, 0.4254 RMSE, and 438.38% MAPE. Figure 5.8 shows the plot between the two methods. We do not experiment with a window size of 1 as the results are known to be poor across all timestep prediction in the previous experiments.

The result of this experiment suggests that recursive multi-step prediction can effectively improve the performance of long prediction timestep without changing the architecture complexity of neural network model. Specifically, the 10 timestep model with a window size of 20 is proven to have the lowest error metrics as compared to other base models in predicting 30 timesteps.

5.7 Summary

In this chapter, we conducted experiments about the trajectory prediction model using AIS time-series dataset and evaluate the performance with 3 defined metrics, MAE, RMSE, and MAPE. Two interpolation techniques, linear and spline, was used and analysed in each experiment. The first three prediction models categorized as direct multi-step prediction were discussed, including 1, 10, and 30 timesteps with a variation of window size each. The last prediction model is also called recursive multi-step where it recursively generates 30 timestep prediction using 1 and 10 timestep base model.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this project, we proposed an analytical landscape to deal with AIS data in Singapore Straits. The work in this project can be split into two parts; first, AIS descriptive-analytic. It includes understanding the underlying of AIS end-to-end communication system, discerning AIS features, and uncovering insight or pattern within the data. The second part is the AIS predictive analytic. The purpose of this part is to develop a predictive model for the vessel future movement. The main challenge of this work is how to give a foundation about AIS analytical discourse in Singapore region, from descriptive-analytic to predictive one. The main contribution of this work is to provide a method, from the methodology, technical library, and future direction, to help researcher investigate deeper into AIS analytic specifically in the Singapore region.

With the current position as the leading maritime capital in the world, Singapore is facing the need to enhance maritime security at Singapore water more than ever. The Singapore Straits is passed by about 100000 vessels each year, and an average of 1 million AIS message is broadcast per day, leading to about 800 messages per minute. AIS message contains valuable information for coastal surveillance, whereby the activity range from vessel monitoring, tracking or directing. The vessel's trajectory prediction model is developed to help monitor maritime activity and hopefully assist the coastal authority for a prompt emergency reaction. AIS exploratory data analysis is discussed beforehand to develop a better understanding of what AIS data can give insight about. The data exploration is accompanied by the statistical description as well as the visualization framework using several built-in tools and libraries.

Trajectory prediction model aims to give the user the future movement of a vessel for a certain timestep given its AIS historical information. We build a data processing pipeline that performs data cleaning, data interpolation, and data extraction. We also design and build two different data input representation for general Machine Learning and Recurrent Neural Network (RNN) model. An archetype of RNN called Long Short Term Memory (LSTM) has an overall better performance metrics as compared to a baseline Linear Regression model. We experiment with 2 interpolation techniques, 3 performance metrics, 3 prediction timestep and several window size. The purpose of the experiment is to analyze under which condition the LSTM model would perform well, the model limitation and the workaround. Prediction model using 1 timestep have high accuracy indicated by a low error performance metrics of MAE, RMSE, and MAPE. However single-step prediction is not useful in practice. Prediction model using 10 timesteps come into rescue with a good balance between prediction accuracy and longer timestep. In this experiment we discovered that MAPE as an error measure have a pitfall causing them to be unreliably high for certain time-series dataset despite the counterpart metric of MAE is low. Prediction model using 30 timesteps is considered inaccurate across all window size, interpolation technique, and most importantly all error performance metrics. One alternative solution without changing the model complexity for this problem is to utilize the 1 and 10 timestep prediction model to predict 30 timestep multiple times using the recursive multi-step technique. The technique is proven to be useful in predicting 30 timesteps without hurting the performance. The best prediction accuracy (lower error) is when the 10 timestep model used to generate 30 timestep prediction. The resulting error metrics go significantly lower from 0.423 MAE to 0.0628 MAE.

6.2 Future Research Directions

As for future work, we would like to leverage big data worth of 1 year of AIS data and use distributed computing framework for data processing in the hope of getting better insight out of AIS data. Another work improvement would be to improve the prediction model by experimenting with more complex architecture and selecting more features combining numerical and categorical data.

We believe the above (but not limited to) future research directions will advance the technology presented in this thesis and contribute to academia and industry.

Bibliography

[1] S. Beveridge, "Least squates estimation of missing values in time series", *Communications in Statistics-Theory and Methods*, vol. 21, no. 12, pp. 3479–3496, 1992.

[2] F. Chollet, "Deep Learning with Python", Manning, Nov. 2017.

[3] T. G. Dietterich, "Machine learning for sequential data: A review", in *Joint IAPR international workshops on statistical techniques in pattern recognition (SPR) and structural and syntactic pattern recognition (SSPR)*, Springer, 2002, pp. 15–30.

[4] M. E. DNV GL, "The leading maritime capitals of the world 2019", *Maritime*, 2019.

[5] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization", *Journal of machine learning research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[6] A. Géron, "Hands-on machine learning with Scikit-Learn and TensorFlow : concepts, tools, and techniques to build intelligent systems", Sebastopol, CA: O'Reilly Media, 2017, ISBN: 978-1491962299.

[7] A. Graser and M. Dragaschnig, "Open geospatial tools for movement data exploration", *KN-Journal of Cartography and Geographic Information*, pp. 1–8, 2020.

[8] A. Harati-Mokhtari, A. Wall, P. Brooks, and J. Wang, "Automatic identification system (ais): Data reliability and human error implications", *The Journal of Navigation*, vol. 60, no. 3, pp. 373–389, 2007.

[9] S. Hexeberg, A. L. Flåten, E. F. Brekke, *et al.*, "Ais-based vessel trajectory prediction", in *2017 20th International Conference on Information Fusion (Fusion)*, IEEE, 2017, pp. 1–8.

[10] R. J. Hyndman and G. Athanasopoulos, "Forecasting: principles and practice", OTexts, 2018.

[11] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy", *International journal of forecasting*, vol. 22, no. 4, pp. 679–688, 2006.

[12] IMO, "Adoption of new and amended performance standards", *Maritime*, 1998.

[13] H. Kang, "The prevention and handling of the missing data", *Korean journal of anesthesiology*, vol. 64, no. 5, p. 402, 2013.

[14] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization", *arXiv preprint arXiv:1412.6980*, 2014.

[15] M. Lepot, J.-B. Aubin, and F. H. Clemens, "Interpolation in time series: An introductive overview of existing methods, their performance criteria and uncertainty assessment", *Water*, vol. 9, no. 10, p. 796, 2017.

[16] S. P. Liraz, "Ships' trajectories prediction using recurrent neural networks based on ais data", Naval Postgraduate School Monterey United States, Tech. Rep., 2018.

[17] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "Statistical and machine learning forecasting methods: Concerns and ways forward", *PloS one*, vol. 13, no. 3, 2018.

[18] M. A. Nielsen, "Neural networks and deep learning", misc, 2018. [Online]. Available: `http://neuralnetworksanddeeplearning.com/`.

[19] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning", *arXiv preprint arXiv:1811.03378*, 2018.

[20] C. Olah, "Understanding lstm networks", `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`.

[21] "Piracy and armed robbery against ships in asia annual report", *ReCAAP*, Dec. 2019.

[22] N. Qian, "On the momentum term in gradient descent learning algorithms", *Neural networks*, vol. 12, no. 1, pp. 145–151, 1999.

[23] E. S. Raymond, "Aivdm/aivdo protocol decoding", https://gpsd.gitlab.io/gpsd/AIVDM.html.

[24] S. RUDER, "An overview of gradient descent optimization algorithms", https://ruder.io/optimizing-gradient-descent/.

[25] S. Shalev-Shwartz and S. Ben-David, "Understanding Machine Learning: From Theory to Algorithms", USA: Cambridge University Press, 2014, ISBN: 1107057132.

[26] "Singapore tops list of leading maritime capitals for fourth time", *Straittimes*, Apr. 2019. [Online]. Available: https://www.straitstimes.com/singapore/transport/singapore-tops-list-of-leading-maritime-capitals-for-fourth-time.

[27] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting", *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[28] "What a singapore strait traffic jam says about the world economy", *SCMP*, Mar. 2019. [Online]. Available: https://www.scmp.com/week-asia/economics/article/2188740/what-singapore-strait-traffic-jam-says-about-world-economy.

[29] B. L. Young, "Predicting vessel trajectories from ais data using r", Naval Postgraduate School Monterey United States, Tech. Rep., 2017.

[30] L. Zhao, G. Shi, and J. Yang, "Ship trajectories pre-processing based on ais data", *The Journal of Navigation*, vol. 71, no. 5, pp. 1210–1230, 2018.