Route Prediction of Tugboats using Deep Reinforcement Learning

2024.MME.8980: Master Thesis

Navneet Sajith - 5432804

Route Prediction of Tugboats using Deep Reinforcement Learning

by

Navneet Sajith - 5432804

Student Name	Student Number
Navneet Sajith	5432804
Report Number	2024.MME.8980

Instructor:	F. Schulte
Secondary Instructor:	M. Saeednia
Project Duration:	Feb, 2024 - September, 2024
Faculty:	Faculty of Mechanical Engineering, Delft

TU Delft

Preface

This thesis marks the culmination of a challenging yet immensely rewarding journey through my master's degree. The road has been long, and I am deeply grateful to those who have supported me along the way.

First and foremost, I would like to express my sincere thanks to my supervisor, Dr. Frederik Schulte. His invaluable feedback, guidance, and unwavering support over the past eight months have been instrumental in the completion of this thesis. Without his insight and encouragement, this achievement would not have been possible.I would also like to extend my heartfelt appreciation to Dr. Mahnam Saeednia for her guidance and mentorship. Her support, especially during moments of difficulty, was crucial in helping me navigate the more complex aspects of this research.

To everyone who contributed to this journey, thank you for your support and encouragement.

Navneet Sajith - 5432804
Delft, September 2024

Nomenclature

Abbreviations

Abbreviation	Definition
AIS	Automatic Identification System
BAP	Berth Allocation Problem
DSR	Design Science Research
ETA	Estimated Time of Arrival
FCFS	First Come First Served
HCC	Harbor Coordination Center
HM	Harbor Master
ICT	Information and Communication Technology
JIT	Just In Time
NC	Nautical Chain
NGO	Non-Governmental Organization
PA	Port Authority
PCO	Port Call Optimization
PCS	Port Community System
PoR	Port of Rotterdam
QC	Quay Crane
QCAP	Quay Crane Allocation Problem
QCSP	Quay Crane Scheduling Problem
STM	Sea Traffic Management
SLR	Systematic Literature Review
VTS	Vessel Traffic Service

Table 1: List of Abbreviations

Symbols

Symbol	Description
f_t	Forget gate activation in LSTM
i_t	Input gate activation in LSTM
c_t	Cell state in LSTM
o_t	Output gate activation in LSTM
W_f	Weight matrix for forget gate in LSTM
W_i	Weight matrix for input gate in LSTM
W_o	Weight matrix for output gate in LSTM
$V_{\pi_{\theta_a}}$	Value function in DRL model
$Q_{\pi_{\theta_a}}$	State-action value function in DRL model
γ	Discount factor in DRL
$A_{\pi_{\theta_a}}$	Advantage function in DRL
E	Encoding of unfinished tasks
o_t	Output probability distribution at time step t
λ	Bias-variance trade-off parameter in GAE
$p_{t,k}$	Output probability of task k at step t
$\hat{y}_t$	Task selected at time step t
u_j^t	Attention score for unfinished tasks in DRL
W_1, W_2	Weight matrices for attention mechanism
v	Parameter vector in attention mechanism
θ_a	Policy parameters in actor network
θ_c	Parameters in critic network
L_{CE}	Cross-entropy loss function
L_{θ_a}	Loss function for actor in DRL model
L_{θ_c}	Loss function for critic in DRL model
smoothL_1	Smooth L1 loss function
s_t	State of the system at time step t
π_{θ_a}	Policy in the actor network

Table 2: List of Symbols

Summary

Efficient Port Operations are essential for minimiming delays and ensuring the safe movement of vessels. Tugboats play a critical role in assisting ships during berthing, unberthing, and navigating port waters. This thesis uses DRL4Route, a deep reinforcement learning (DRL) framework aimed at optimising tugboat routes and pick-up locations. By using historical towage data and adapting to the dynamic conditions of the Port of Rotterdam, DRL4Route provides real-time recommendations that streamline tugboat operations, reduce delays, and improve safety.

The core of the DRL4Route framework lies in its ability to continuously learn and adapt to real-world conditions. Unlike traditional static models, DRL4Route leverages spatio-temporal data to predict optimal tugboat routes. This allows for real-time evaluations that can help the predicted route match closely with the actual route which helps tugboats arrive at the right location at the right times. The framework's focus on optimising both pick-up and drop-off points helps port operators avoid inefficiencies that can arise from poorly coordinated tugboat movements.

By improving the efficiency of tugboat operations, DRL4Route contributes to a more sustainable and resilient port ecosystem. The system's adaptability and potential for real-time decision-making make it a strong candidate for future automation of tugboat operations. This thesis highlights how advanced machine learning techniques can enhance the performance of ports like Rotterdam, driving economic benefits and reducing the environmental impact of maritime operations.

Contents

1

Introduction

Ports play a crucial part in the facilitation of global trade and are responsible for the transport of approximately 80% of global trade [17] and ensure the safe transshipment of goods from ships to land and vice-versa [54]. Recent challenges such as the COVID-19 pandemic and instability caused by the wars in Ukraine and Palestine have shaken the industry but it still stands strong with a 2.4% projected increase in trade in 2023 and a 2% increase annually from 2024 to 2028 [14]. The Port of Rotterdam is Europe's largest port and it moves over 438 billion tons in cargo and adds €38.6 billion in annual trade which equates to 3.2% of the Dutch GDP [30]. This volume of cargo, the breakup of which can be seen in Figure 1.1, shows the importance of efficient port call optimsation to ensure this massive and intricate machine, that is the port, functions optimally to facilitate the movement of all the cargo and ships within it. The planned vessel movements are essential in this system as they balance time, allocate resources, and ensure safe operations in the port. Optimising port call processes are necessary to minimise delays which can occur due to uncertain weather conditions, dynamic vessel schedules and possible infrastructure limitations.

(Gross weight x 1,000 tonnes)	2023	2022	Difference (number)	Difference (%)
Dry bulk cargo	70,642	80,064	-9,422	-11.8%
Liquid bulk cargo	205,627	212,771	-7,144	-3.4%
Total bulk cargo	**276,269**	**292,835**	**-16,566**	**-5.7%**
Containers	130,162	139,657	-9,495	-6.8%
Break bulk	32,371	34,889	-2,518	-7.2%
Total general cargo	**162,533**	**174,546**	**-12,013**	**-6.9%**
Total cargo throughput	**438,802**	**467,381**	**-28,579**	**-6.1%**
Total numbers of containers	7,816,755	8,315,417	-498,662	-6.0%
Total TEUs	13,446,709	14,456,313	-1,009,604	-7.0%

Figure 1.1: Throughput in Port of Rotterdam [30]

Figure 1.2 shows the complex coordination required during a ship's port call and how the multiple actors in the system collaborate to provide a smooth arrival, berthing, and departure process. Each coloured line corresponds to a different service provider or actor involved in the process such as the vessel, port authority, pilot, tugboat, mooring crew, and terminal. The horizontal timeline shows the key events and stages which range form the ship's arrival to the port or anchorage, through berthing and operational

activities, and finally its departure from the port. Intersections between the lines show the critical points where the actors have to coordinate carefully with one another such as the point where the tugboats are used or securing the ship at berth and these events must be synchronised between different parties. The diagram highlights the need for real-time data sharing and precise timing to avoid delays, reduce waiting times, and optimise the overall port call process.

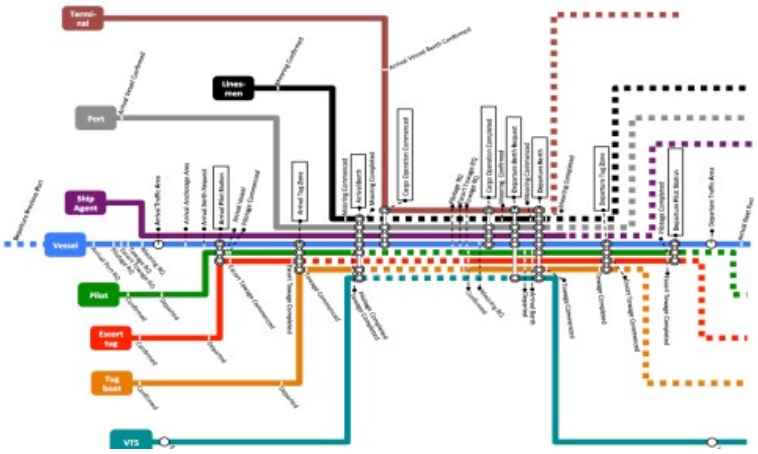

Figure 1.2: Port Call Metro Map [24]

Ships need the tugboats provided by towage companies to safely navigate the narrow and shallow waters of the ports [18]. The quantity and duration of this assistance is decided by the pilot who oversees the navigation of the ship [28]. These towage operations are usually handled by private organisations because it has been observed that this drives more competition and improves the quality of the assistance provided to the ships.
Tugboat companies like Kotug International BV play a vital role in safe vessel operating within the port. KOTUG provides several services which include the designing, building, chartering, and operating vessels, training people and also providing consultancy services worldwide. In the Port of Rotterdam, KOTUG International B.V offers towing services to ships entering and leaving the harbour. Kotug assigns tugboats that assist large sea-going vessels to operate safely within the port and with berthing and unberthing. Tugboats have been used in this role for decades, with the first commissioned tugboat dating back to 1842. In a port such as Rotterdam, which handles approximately 30,000 sea-going vessels annually, tugboats are priceless for the service they provide. KOTUG is constantly seeking innovative solutions to further enhance the efficiency of their operations and bring down emissions from their operations. Their challenge lies in coordinating tugboat movements within the complex network of activities at the Port of Rotterdam to best service their customers while keeping operational costs to a minimum.

The Port of Rotterdam does not have the standardised regulations that determine the location of each vessel. For this reason, tugboats are sometimes connected with tugboats in locations that are different from the scheduled locations. Efficiently and safely scheduling activities for a large number of vessels presents significant challenges and accurately predicting the locations where vessels are connected to and disconnected from tugboats is vital for optimising port throughput and enhancing the competitiveness of both the port and the company on a global scale.

Figure 1.3: Tugboats guiding a vessel [20]

The operation of tugboats in the port can be seen as a pick-up and delivery service problem. It has been observed that the actual routes taken by the pilots often differ from the best routes that routing tools have predicted [3] [45]. A study that conducted experiments in the US and Mexico for a large soft drinks company showed that three out of four deliveries deviated from the planned route [23] [6]. Accurate route prediction can reduce these differences between the predicted and actual routes.

This thesis explores novel methods for optimising KOTUG's tugboat pick-up optimisation and route prediction. The core concept DRL4Route is a deep reinforcement learning framework designed to optimise pick-up locations and route planning. DRL4Route [8] uses neural networks to learn and adapt to dynamic port environments, offering significant potential for improved tugboat efficiency and effectiveness.

Route prediction plays a significant role in improving the efficiency of operations in a port. Accurate route prediction can allow for superior resource allocation, reduce operational costs, and improve safety standards by minimising the distance between the predicted and actual routes. Focusing on tugboat operations, accurate route prediction will enable the safe navigation and berthing of large vessels. Discrepancies between predicted and actual routes can often lead to increased fuel consumption, higher emissions, and overall increased inefficiencies.

The method of route prediction used in this thesis is DRL4Route which is a type of Deep Reinforcement Learning (DRL). DRL is an advanced machine learning method that combines Deep Learning (DL) and Reinforcement Learning (RL) to generate the best decisions possible in complex environments [36]. DL uses neural networks with multiple layers to model, identify, and understand intricate patterns in the given data. This can be very useful in route prediction as Deep Learning can identify trends in the data that can facilitate better decision-making for the route prediction. RL trains agents to perform tasks by evaluating the performance of the system at every time step and providing feedback based on the evaluation metrics used in the system. This feedback is given in the form of rewards based on the performance. The goal of the system is to maximise the overall cumulative reward [21].

Figure 1.4 shows the framework of the DRL4Route method, which comprises an Actor-

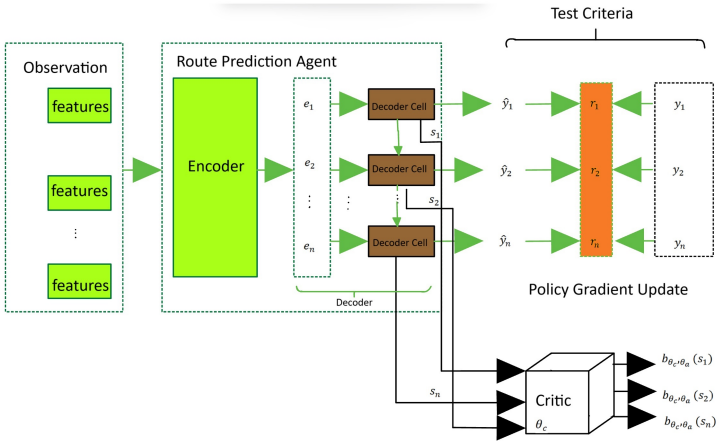

Figure 1.4: Framework of DRL4Route

Critic architecture, which is the reinforcement learning method used. There is an observation phase where the features are introduced into the model as input which is then processed by the route prediction agent. This agent is made up of an encoder that converts the input features into encoded representations and a series of decoder cells that generate the actual route predictions. The decoder outputs are then compared against the actual route that is followed to compute the rewards. These rewards, along with the state information that is passed to a critic network, are used to perform a policy gradient update which is used to make the route prediction agent learn how the model works in order to make more accurate predictions through reinforcement learning.

DRL is very useful in dynamic environments and it can learn optimal routing strategies by observing and interacting with the environment. It is constantly learning which means that it will only get better the longer it is left in the system. DRL has been shown to optimise routing strategies by automatically adapting to prevailing traffic conditions and suggesting routing strategies that can minimise delays in the system [35]. DRL can be scaled to be useful in large and complex networks. This property of DRL makes it ideal to be used in environments such as the port of Rotterdam which has a large number of moving parts and situations which affect the state of the system. The following case studies have been developed using DRL and show preferable results.

- Trailnet for IP Networks [35]:
  Trailnet is a DRL method for routing in IP networks which replaces forwarding tables with a computational model that is trained using value iteration and stochastic gradient descent. The function of this model is to estimate the cost of IP packet forwarding along different ports and selects the best port that minimises the given cost function. This model showed a very high accuracy and fast interference times when it was tested on large network topologies.
- SDN Routing Optimisation [36]:
  DRL has been used to optimise routing in software-defined networks where it analyses the traffic conditions and develops strategies that minimise delays and improves throughput which results in improved performance that outperforms the traditional optimisation algorithms.

This research aims to develop an improved route prediction method and optimise

pick-up locations and route planning using advanced algorithms and machine learning. The expected outcomes of this research are increased operational efficiency, reduced costs and to provide Kotug with the information and methods to navigate the extremely busy waterways of the port of Rotterdam.

This thesis has the potential to advance the tugboat operations of KOTUG and pave the way for additional research opportunities. Additionally, the hope is to transform the current challenges into opportunities and pave the way to a more efficient, sustainable, and resilient port ecosystem.

2

Literature Survey

Port call optimisation is critical to advancing maritime logistics, improving the efficiency and effectiveness of processes involved in managing vessels entering and exiting ports. As global trade volumes increase, ports like Rotterdam are tasked with handling a growing number of ships, making it crucial to optimise operations such as berth allocation, tugboat scheduling, route prediction, and fleet management. This section reviews recent port call optimisation advancements, focusing on pick-up and route prediction methodologies that could benefit tugboat operations at the Port of Rotterdam. The review categorises relevant studies into different thematic areas and highlights their contributions and relevance to the study.

Category	Paper References
Berth Allocation, Scheduling, and Port Infrastructure	Rodrigues and Agra [33], Conca et al. [5], Hendriks et al. [13],Gharehgozli et al.[10],Zhen et al.[53]
Route Prediction and Optimisation	Wang et al. [42], Cho et al. [4], Yang et al. [16], Du et al. [7], Mao et al. [25], DeepRoute+, Qian [32], Wen [45], FDNet [8], Graph2Route Wen [44], OSquare Zhang [52], Wu et al.[48]
Fleet Management and Operational Efficiency	Wu et al. [47], Li et al. [22], Rodrigues and Agra [33], Merkel et al. [26], Poulsen and Sampson [31]
Tugboat Scheduling	Wang et al. [41], Wei et al. [43], Wang et al. [39], Wang et al.[40], Yao et al.[49], Yu.[50]

Table 2.1: Categorisation of Papers in Port Call Optimisation

Table 2.1 highlights the various papers consulted for this study, organising them into categories relevant to port call optimisation. The table helps illustrate how different aspects of port operations can contribute to optimising processes, particularly concerning the scheduling and allocation of resources in ports.

Berth Allocation, Scheduling, and Port Infrastructure

The optimisation of berth allocation, scheduling, and port infrastructure is a crucial component of port call optimisation. Berth allocation plays a vital role in ensuring that ships can dock efficiently and with minimal delays and disruptions.

Rodrigues and Agra [33] investigate berth allocation and quay crane scheduling under uncertainty, which are essential components of port call optimisation. Their research

addresses how uncertainty in ship arrival times can complicate the planning process, and proposes strategies for mitigating the impact of such uncertainties. By optimising berth allocation, the port can increase throughput and reduce waiting times for incoming vessels, which directly improves the overall efficiency of port operations. Conca et al. [5] explore real-time data sharing and automation in port call processes, which could significantly enhance tugboat operations. By using real-time data from various sources—such as vessel tracking systems and automated terminal systems—port authorities can make informed decisions on the scheduling of tugboats, cranes, and other port resources. This timely exchange of information between different stakeholders (e.g., shipping companies, terminal operators, and tugboat service providers) can lead to more coordinated operations, reducing delays and increasing the efficiency of the entire process. Hendriks et al. [13] delve into strategic investments in port infrastructure, which are critical for the long-term sustainability of ports. Their work suggests that optimising infrastructure investments, such as expanding berths or modernising equipment, can have a significant impact on reducing bottlenecks and enhancing operational efficiency. This is particularly relevant to ports like Rotterdam, which handle a large volume of container traffic and require constant upgrading to meet the demands of modern shipping. Gharehgozli et al. [10] and Zhen et al. [53] further contribute to this field by examining the integration of sea and land-side operations. Their studies highlight the importance of synchronising port infrastructure with hinterland logistics, ensuring that goods can be efficiently moved in and out of ports. This synchronisation is crucial in reducing congestion, minimising idle times for vessels, and streamlining the overall port call process.

Route Prediction and Optimisation

Route prediction and optimisation are central to improving the efficiency of maritime logistics, particularly in dynamic environments like the Port of Rotterdam, where multiple ships, trucks, and tugboats must operate in tandem. Advances in this area can significantly reduce waiting times and improve the allocation of resources such as tugboats, which are essential for guiding large vessels through narrow port channels.

Wang et al. [42], Cho et al. [4], and Yang et al. [16] explore optimisation models for liner container shipping routes, offering insights into how network design and scheduling decisions can affect overall efficiency. These models, although focused on large container shipping, provide valuable methodologies that could be adapted for tugboat route optimisation, particularly in complex, congested ports. Du et al. [7] propose a machine learning-based approach for liner shipping schedule design, which integrates predictive models for traffic patterns and port congestion. This approach could be adapted for predicting tugboat movements within ports, offering a way to anticipate delays and optimise the allocation of tugboat resources based on real-time conditions. One of the key papers in this area is by Mao et al. [25], which presents a deep reinforcement learning framework, DRL4Route, tailored specifically for pick-up and delivery route prediction. This framework has significant relevance to optimising tugboat routes in dynamic environments like the Port of Rotterdam. DRL4Route offers a flexible approach to route optimisation, allowing for real-time adaptation to changing conditions, such as varying weather patterns or fluctuating traffic within the port. The reinforcement learning aspect also enables the system to learn from previous decisions, progressively improving the accuracy and efficiency of tugboat route predictions over time. The DeepRoute+ model discussed by Wen [45] takes a similar approach

but applies it to the courier industry. This deep neural network model is designed to predict courier pick-up routes by taking into account spatial-temporal constraints and individual courier decision preferences. The model can be adapted to maritime logistics, particularly in predicting tugboat movements, by considering constraints such as ship arrival times, tide conditions, and tugboat availability. The model's ability to adapt to individual preferences (in this case, the decision-making preferences of tugboat operators) adds an extra layer of flexibility to its application in real-world scenarios. Qian [32] offers Basic Time Greedy and Basic Distance Greedy models, which provide a simple yet effective method for route optimisation by focusing on minimising time and distance, respectively. These models can be applied to tugboat route planning by ensuring that tugboats follow the shortest or quickest routes, depending on the specific goals of the operation (e.g., minimising fuel consumption or maximising the number of completed tasks). FDNet [8] is another deep learning model that leverages spatio-temporal features for route prediction. Similar to DeepRoute+, it uses dynamic data to predict routes, offering potential applications for scheduling and optimising tugboat operations. Graph2Route, proposed by Wen [44], builds on this by using dynamic spatio-temporal graphs, which could be especially useful in ports where conditions change frequently, such as Rotterdam. By utilising these graphs, port authorities can predict the most efficient routes for tugboats, even in the face of unpredictable factors like weather or ship traffic. OSquare, developed by Zhang [52], uses machine learning techniques, specifically XGBoost, to optimise routes. The system has been applied to instant delivery services, but its core principles—such as learning from previous route decisions and adjusting dynamically—could be adapted for use in tugboat scheduling and route optimisation. OSquare's ability to predict route efficiency based on real-time data inputs makes it an attractive option for complex logistical environments like ports.

Fleet Management and Operational Efficiency

Fleet management and operational efficiency are key areas of focus in port call optimisation, as ports must manage large numbers of vessels and ensure that resources are allocated effectively.

Wu et al. [47] present a comprehensive study on fleet deployment, refueling strategies, and speed optimisation within liner shipping networks. The paper emphasises reliability and travel time efficiency, both of which are critical for effective port operations. Their methodology could be adapted to manage the tugboat fleets in large ports like Rotterdam, ensuring that resources are deployed in a way that minimises delays and maximises fuel efficiency. Li et al. [22] focus on the optimisation of fuel consumption and cost reduction for tugboats. The paper uses predictive models to determine engine power requirements, which could directly impact the operational efficiency of tugboats in the Port of Rotterdam. By optimising fuel consumption, the port could not only reduce operational costs but also contribute to environmental sustainability by cutting down on emissions. Merkel et al. [26] and Poulsen and Sampson [31] explore the connection between port call optimisation and greenhouse gas emissions. As environmental regulations become stricter, ports are under increasing pressure to reduce their carbon footprint. Optimising fleet management and scheduling can play a significant role in this effort, as more efficient operations lead to reduced fuel consumption and emissions. Rodrigues and Agra [33] again come into play here, as their work on berth allocation and crane scheduling under uncertainty directly impacts fleet management. By ensuring that ships are allocated berths efficiently and that cranes

are deployed at the right time, ports can improve the overall flow of goods and reduce idle times for both ships and tugboats.

Tugboat Scheduling
Tugboat scheduling is a critical part of port operations, as tugboats are essential for guiding large vessels through crowded and often narrow waterways. Efficient scheduling of tugboats ensures that vessels can enter and exit ports without unnecessary delays, contributing to overall port efficiency.

Yao et al. [49] propose an Improved Grey Wolf Optimisation (IGWO) algorithm for solving multi-objective tugboat scheduling problems. The model considers several key objectives, including tugboat operational time, fuel consumption, and efficiency, making it particularly suited for large, busy ports like Rotterdam. The IGWO algorithm improves upon previous optimisation techniques by introducing enhanced convergence parameters, allowing for more accurate solutions to complex scheduling problems. Yu [50] focuses on the development of a mixed-integer linear programming (MILP) model for tugboat scheduling, with a case study centered on Singapore Port. The MILP model aims to minimise processing costs by optimising the allocation of tugboats to incoming vessels. The model's application to a real-world scenario like Singapore, one of the busiest ports in the world, demonstrates its effectiveness in managing congested port environments. This model could be adapted for use in Rotterdam, where similar challenges are faced. Other works, such as those by Wang et al. [41] and Wei et al. [43], also address the tugboat scheduling problem, though they focus more on optimising individual components of the scheduling process, such as minimising tugboat idle times or maximising operational efficiency. While these studies provide useful insights, they do not fully address the predictive route optimisation approach emphasised in this study.

The reviewed papers offer a deep understanding of the field of port call optimisation, with specific focus on pick-up and route prediction methodologies that can enhance tugboat operations. As ports like Rotterdam continue to grow, the need for efficient scheduling and route prediction becomes increasingly important. The studies reviewed here showcase the potential of modern machine learning techniques, optimisation algorithms, and predictive models to transform how ports manage their resources, reduce costs, and improve overall efficiency.

By integrating insights from these studies—such as predictive models for route optimisation and multi-objective scheduling algorithms—port authorities can significantly enhance the operational efficiency of tugboats, leading to faster turnaround times, lower fuel consumption, and reduced emissions. The applicability of these models to real-world scenarios, such as the Port of Rotterdam, highlights the importance of continued research and development in this area, as well as the potential for significant improvements in the way ports manage their operations.

2.0.1. Research Proposal

Based on the literature survey and the data available from Kotug, this thesis works to answer the question *"How can historic tugboat towage data be used to optimise pick-up locations for tugboats and predict their routing within the Port of Rotterdam?"*
This question can be split into two parts:
Routing Prediction:

- How can historical tugboat towage data be used to develop accurate and reliable routing prediction models?
- What predictive models or algorithms can be developed to forecast tugboat routing from identified pick-up locations?

Verification and Validation:

- What metrics and criteria should be used to evaluate the effectiveness and efficiency of the proposed models?
- How do the proposed models compare to current practices regarding operational performance?

3

Problem Definition

The towage process within the Port of Rotterdam is essential for ensuring the safe and efficient movement of vessels, which involves three key types of movements: berthing, unberthing, and shifting. This research focuses on the first two movements, berthing, and unberthing, which are critical for maintaining the smooth operations of the port. Towage plays a central role in these processes, as vessels, especially larger ones with significant propulsion requirements, often need assistance from tugboats to navigate safely within the port due to water depth limitations and geometric constraints of berths. Additionally, environmental factors and potential interactions with other vessels require precise control over speed and course which underlines the importance of tugboat assistance to prevent accidents or collisions.

The current methods for assigning the route of the tugboats depend on human expertise to understand the optimal route. The Vehicle Routing Program optimises the system before the pick-up and delivery processes but lacks the flexibility for real-time adjustments. These programs assume static conditions and may not adapt well to real-time changes in the environment such as vessel breakdowns or last-minute delivery change requests. These issues can make it challenging for the system to work in environments with spontaneous changes in work conditions [12] [19]. The complexity of the system also plays a major role in the overall process and as the complexity of the system increases, a lot of VRP algorithms can lose their effectiveness drastically which could lead to delays and increased costs [51].

The towage process starts well before a vessel arrives at the port. The vessel informs the Harbour Master 24 hours in advance, after which the pilot and tugboats are scheduled based on the vessel's size and operational needs. Six hours before reaching the pilot boarding point, the piloting organisation, in this case KOTUG, is alerted of the ship's expected arrival. Then, at 2.5 hours before reaching this location, the pilot is officially scheduled and requested. At this stage, a berth availability check is conducted. If no berth is available, the ship must anchor outside the port, and the pilot service is canceled, which can be done without penalty up to one hour before the ship reaches the boarding location. Once a berth is allocated by the terminal, the ship is cleared to proceed, and if necessary, the pilot is rescheduled and boards the vessel at the designated location. Once both the tugboats and pilot are ready, the mooring pro-

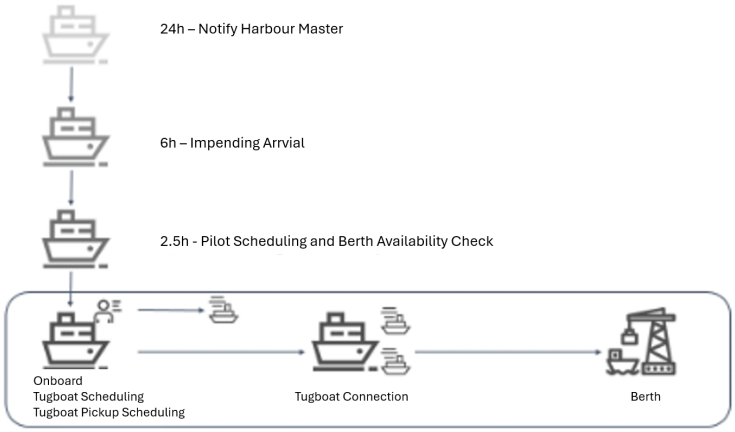

Figure 3.1: Tugboat pick-up process

cess begins, allowing the terminal to start its operations and marking the conclusion of the berthing process [38]. However, predicting the exact locations where vessels should connect with or be released from tugboats remains a challenge despite extensive scheduling. KOTUG is one of the companies responsible for managing these tugboat operations and it uses a scheduling tool called Kotug Optiport that predicts pick-up and drop-off locations. However, discrepancies between predicted and actual towage locations frequently occur as these locations are ultimately determined by pilots based on real-time conditions. This misalignment can lead to inefficiencies such as delays, increased fuel consumption, and suboptimal resource utilisation. The towage process can be observed in Figure 3.1

The core problem lies in the difficulty of precisely predicting these connection/disconnection locations in advance. The dynamic nature of the port environment, coupled with the varying sizes and navigation capabilities of vessels, makes it challenging to allocate tugboat resources efficiently. These prediction errors can result in significant operational issues, including increased waiting times, delays in terminal operations, and reduced overall port efficiency. From an economic standpoint, these inefficiencies translate into higher operational costs and lower profitability for both KOTUG and the Port of Rotterdam.

Given these challenges, there is a pressing need for more accurate and dynamic predictive models to optimise the scheduling and routing of tugboats. These models should be able to predict connection and disconnection points more reliably, using historical towage data and incorporating real-time adjustments to mitigate the inherent uncertainties of port operations. By mimicking the decision-making strategies of experienced pilots and dynamically adjusting to new tasks or conditions, these models would pave the way for automated and more efficient tugboat operations.

The research proposed in this study centers around developing a predictive model, DRL4Route, which leverages Deep Reinforcement Learning (DRL) techniques to outperform existing baseline methods for route prediction. This model aims to offer real-time flexibility, improve the precision of pick-up/drop-off location predictions, and optimise tugboat usage. By improving the accuracy of these predictions, the model will not only enhance the efficiency of towage services but also contribute to reducing delays, lowering operational costs, and improving the overall economic performance of

the Port of Rotterdam.

The goal of this research is to address the challenges in predicting towage locations and optimising tugboat schedules through the integration of advanced machine learning techniques. This solution seeks to improve operational efficiency, ensure the safety of vessels and infrastructure, and drive economic benefits for both the tugboat operators and the port as a whole.

4

Methodology

This thesis tests the effectiveness of different route prediction techniques for tugboats in the port of Rotterdam. This process involved looking at various methods of route prediction and evaluating the best possible method to adopt for this model. The following models have been evaluated for this thesis: Basic Time Greedy model[32], Basic Distance Greedy [32] model, Deeproute [45], FDNet [8], Graph2route [44], OSquare [52], and DRL4Route [25]. The roles involved in the pickup and dropoff of the ships can be seen as the ship giving the information that it has arrived at the port or is ready to depart, the tugboats picking the ships, and the tugboat finishing the task assigned to it. This project is focused on predicting the route of the tugboat given its unfinished tasks. These unfinished tasks can be defined as shown in Equation 4.1 where *n* is the number of unfinished tasks at the query time, o_i is the i-th task associated with

$$O^w = \{o_i \mid i = 1, \ldots, n\} \tag{4.1}$$

These unfinished tasks can be mixed and matched to produce an order of unfinished tasks. This order is called a service route as shown in Equation 4.2 where $y_j \in \{1, \ldots, n\}$, and if $j \neq j'$ then $y_j \neq y_{j'}$.

$$Y_w = (y_1, \ldots, y_n) \tag{4.2}$$

Finally, there are route constraints that are used to ensure real-world conditions such as the agents in the system need to conduct pick-ups before the delivery process [8] and capacity constraints which in this case is 1 because a tugboat can only service one ship at a time. These constraints can be represented by *C*. The problem statement of this model which is to predict the tugboat driver's (w) decisions of the unfinished task set O^w which abides by all the route constraints *C*. Equation 4.3which

$$F_C(O^w) = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_m) \tag{4.3}$$

4.1. Data and Preprocessing

The data used for this thesis has been provided by KOTUG. This data shows the vessels and tugboat operations between the 31st of May 2022 and the 31st of May, 2023. As this thesis focuses on the route prediction of tugboats, the dataset containing the

data of all the tugboat operations is looked at. The original dataset is provided in Excel format and includes extensive records of the tugboat movements in this timeframe. The dataset contains 42,232 entries across 15 columns which include both numerical and categorical data that correspond to tugboat movements. The data preprocessing steps are aimed at cleaning the data, handling missing values, and preparing it for further analysis. The first step involves handling missing values. The data that corresponds to the Berth locations have 20,729 missing entries in the column ”From Berth” and 16,336 missing entries in the ”To Berth column. Also, the ”From Haven” column has 20,807 missing entries and the ”To Haven” column has 16,491 missing entries. These missing entries have been filled with a placeholder text ”Unkown” to retain the record without losing any information. Furthermore, the columns containing the dates in the original dataset are converted to a standard ”datetime” format that allows for time-based operations to be performed using the dates. This conversion is essential for calculating the duration of operations and analysing the time trends in the system. The columns containing the geographical locations of the tugboats ”From Location ”,”From Location Y”, ”To Location X”, and ”To Location Y” were checked for consistency and standardisation so that these could be used for the route prediction problem. These were found to be correctly formatted floating-point numbers and did not require and additional standardisation. These were the preprocessing operations conducted to ensure that the dataset contained no missing values, and for all the columns to contains standarised data. This prepared the dataset for the subsequent route prediction problem.

4.2. Encoder-Decoder Architecture

The Encoder-Decoder architecture [29] is a popular framework used in several machine learning applications. This framework can handle route prediction tasks for tugboats in the Port of Rotterdam. The Encoder-Decoder architecture is adept at handling sequence-to-sequence tasks, making it ideal for handling tasks in the route prediction field.

The model is split into the Encoder and the Decoder components. The Encoder processes the historical route data and current state of the tugboats and places this data into a fixed-size context vector. This context vector understands the state of the system input based on the recent trajectory and operational context. The Encoder consists of layers that handle sequential data such as the Long Short-Term Memory (LSTM), Grated Recurrent Units (GRU), and Transformers. The Encoder input is usually the past locations of the tugboat and possibly additional features such as timestamps, speeds, and headings of the tugboat. With each iteration of the sequence, the Encoder updates its internal state and, finally, generates a context vector that contains all the necessary information for accurate route prediction.

After the Encoder process, the Decoder then uses the context vector to generate the future routes of the tugboat. It uses the current state of the system and the context vector to predict the future steps in a step-by-step process. The Decoder is similar to the Encoder in structure and has layers such as LSTM, GRU or Transformers. This iterative process continues until the entire route is generated.

To use the Encoder-Decoder architecture for route prediction models, the data needs to be prepared. Historical route data with timestamps, GPS coordinates and other

relevant features must be collected and preprocessed. This preprocessing ensures that the data is a suitable input. The next step is the training of the model. The data is split into testing, training, and validation sets. The architecture of the Encoder-Decoder is defined using the necessary layers and parameters. The model is trained by feeding it historical data and using a learning rate to improve the training efficiency.

The model is then evaluated to check the parameters and avoid overfitting. This process involves validation to check the performance and make adjustments. Finally, the model is tested again with some test data to evaluate the prediction model with the existing model using several metrics discussed later.

The Encoder-Decoder architecture has numerous advantages for route prediction. The architecture allows the system to use various features apart from just location such as speed, direction, weather, and port traffic. The ability to handle sequential data makes it very good at tasks where past locations influence future movements. The scalability of this architecture allows it to handle large datasets and complex prediction tasks.

4.3. DeepRoute

DeepRoute is an advanced neural network model that is used to predict package pick-up routes of couriers by analysing the information gathered from their spacio-temporal behaviours. This model is exceptional for optimisation of logistics operations, efficient package dispatching, accurate arrival-time estimations, and risk management of delivery delays. There are several layers which together make up Deeproute.

4.3.1. Spatio-Temporal Encoder

This Encoder, also called the Transformer Encoder, is a component of DeepRoute that processes and encodes the spacio-temporal features of the data provided. It uses a multi-head attention mechanism to capture complex dependencies within the input data. The multi-head attention ensures that the model simultaneously focuses on various sections of the sequence. This property of the Encoder helps the system detect and learn numerous patterns. The encoder consists of sequentially structured linear and multi-head to process input features. Input data is passed through these layers in the forward pass. The attention mechanisms assist in understanding the importance of the different parts in the sequence. The encoder output is a set of context vectors that show the spacio-temporal information of the couriers.

4.3.2. Decoder with Attention Mechanism

The role of the Decoder is to use the LSTM cells and attention mechanisms to generate the predicted sequence of locations. Its main task is to use the encoded context vectors to provide a sequence of actions. The attention mechanism within the decoder makes the model focus on relevant parts of the context vectors during each decoding step. The decoder is initialised with LSTM cells, attention modules, and softmax layers. The decoder makes use of a recurrence function to update the hidden states and generates the predictions at each step. This recurrence function works with the attention mechanism to simultaneously focus on sections of the context vector.

4.3.3. Route Prediction and Masking Techniques

DeepRoute uses masking techniques to handle route constraints and for the predicted routes. These techniques are used to make the model adhere to the spatial and temporal constraints of the environment. Some of these constraints are time constraints and geographical barriers which these masking techniques can account for. The recurrence function in the decoder updated the masks based on the previous predictions. This means the model does not allow the courier to visit previous locations.

4.3.4. Mathematical Model

Attention Mechanism:

The attention mechanism of the DeepRoute model for the tugboat route prediction calculates a weighted sum of encoder outputs based on the relation between the current decoder hidden state and ecnoder outputs.

$$u_{i,j} = v^T \tanh(W_{\text{query}} h_i + W_{\text{ref}} h_j) \tag{4.4}$$

W_{query} and W_{ref} are weight matrices, v is a parameter vector, h_i is the hidden state of the decoder at time step i, and h_j is the hidden state of the encoder at time step j. This equation makes the model focus on all parts of the input sequence.

This process involved the analysis of a dataset containing information about the tugboat operations with values such as the tugboat ID (IMO and MMSI), its current and required locations (with coordinates), the berths, havens, and names of these locations. This dataset contains the following columns: IMO, MMSI, Name, From, To, From Location X, From Location Y, To Location X, To Location Y, From Berth, To Berth, From Haven, To Haven, From Location Name, and To Location Name. These values provide the necessary information of tugboat's movements required for the route prediction process.

Multi-Head Attention:

This module gives the system the ability to run through the attention mechanism several times in parallel. This is defined Equation 4.5 where each head is given by Equation 4.6. These equations allow the system to go through various sequences parallelly in the input data.

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h) W^O \tag{4.5}$$

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{4.6}$$

In the codebase, the DeepRoute class uses the encoder and decoder to predict the tugboat routes. The enc_sort_em function encodes the input features and prepares the initial states for the decoder. It generates attention masks and processes the input through the transformer encoder, resulting in the decoder's context vectors and initial hidden states.

The input data 'V' and reachability masks V_reach_mask are processed to generate the predicted routes in the forward pass. The decoder iterates through the generated sequence to update its states and makes predictions at each step. The outputs are the log probabilities and the selected routes which result in the prediction of the tugboat's path.

4.4. FDNet

Feauture-based Dynamic Network (FDNet) is a neural network model designed to predict package pickup routes by using spacio-temporal features. This model works to optimise logistics operations by accurately forecasting the sequence of locations a tugboat will visit. FDNet uses techniques like LSTM encoders, attention mechanisms, and multi-layer perceptors to find a solution.

The architecture of FDNet is given below:

- spacio-temporal Encoder: The LSTM encoder within FDNet processes spacio-temporal input features and converts these into a fixed-size representation. The encoder uses a bidirectional LSTM to understand forward and backward dependencies in the input sequence.
- Decoder with Attention Mechanism: The Decoder generates the predicted sequence of locations based on the encoded input features. An LSTM cell with attention mechanisms is used to find the important parts of the input sequence at each step. The decoder can operate in different modes such as greedy search and beam search to generate varied predictions.
- Time Prediction Module: The Time Prediction module predicts the time duration between tugboat stops. It combines feature embeddings, factorisation machines, and multi-layer perceptors to estimate the time required per section of the route. This module performs the temporal aspect tasks of the route prediction.
- Feature Update Mechanisms: FDNet has mechanisms for updating features based on the current state. The tp_update_features' and 'rp_update_features' functions regularly update the input features at each step for accurate predicitions.

4.4.1. Mathematical Model

FDNet uses complex formulae which make it an adept model for route predictions.

Attention Mechanism:

The terms used in the given equations are:

- W_{query} is a weight matrix for the query.
- Conv1D is a 1D convolution applied to the reference.
- v is a parameter vector.
- Q' and R' are the projected query and reference vectors, respectively.
- C is a scaling factor.
- h_i are the hidden states of the input sequence.
- tanh is a hyperbolic tangent function that maps real numbers in the range -1 to 1.

Query and Reference Projections:

$$Q' = W_{\text{query}} Q \tag{4.7}$$

$$R' = \text{Conv1D}(R) \tag{4.8}$$

Equation Equation 4.7 multiplies the query vector with the weight matrix and Equation 4.8 is used to convert the reference matrix using a 1D convolution.

Compatibility Scores:

$$U = v^T \tanh(Q' + R') \tag{4.9}$$

Logits Calculation:

$$\text{logits} = \begin{cases} C \cdot \tanh(U) & \text{if use\_tanh is True} \\ U & \text{otherwise} \end{cases} \tag{4.10}$$

Attention Weights:

$$\alpha = \text{softmax}(\text{logits}) \tag{4.11}$$

Context Vector:

$$c = \sum_i \alpha_i h_i \tag{4.12}$$

LSTM Encoder:
The LSTM encoder processes input sequences and generates hidden and cell states:

LSTM Equations:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{4.13}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{4.14}$$

$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{4.15}$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \tag{4.16}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{4.17}$$

$$h_t = o_t * \tanh(c_t) \tag{4.18}$$

σ is the sigmoid function, and $*$ denotes element-wise multiplication. The LSTM uses gates (forget gate, input gate, output gate) to control the flow of information and update its cell and hidden state.

After this step, Recurrence mechanism is used to update the hidden states and calculate the logits for the next steps. This involves the use of a Beam search which is a way to hold onto the best solution after each iteration. Following this step, a masking mechanism filters out the unusable sequences. The system then uses the final data and initialises the embedding dimensions, hidden state dimensions and other parameters for a complete end-to-end learning structure. **Recurrence Mechanism:**
The recurrence mechanism updates the hidden states and calculates logits for the next steps:

The FDNet class uses the encoder, decoder and time prediction modules for route prediction. The input is passed through the encoder to gain the context vectors in the forward pass. The decoder then generates a predicted route for the tugboats from these context vectors. The time prediction module predicts the duration between each successive stop of the predicted route. Using these methods, FDNet can be used successfully for route prediction purposes.

4.5. Graph2Route

Graph2Route is a dynamic spacio-temporal graph neural network developed to predict pick-up and delivery routes. It optimises the prediction accuracy by using the graph structure of tasks along with the spacio-temporal features. Graph2Route solves the problems by looking at the tasks as graphs. This method allows it to identify different tasks over time and space and, therefore, gives it a better grasp of the route prediction problem.

4.5.1. Graph2Route Architecture

Graph2Route uses a spacio-temporal graph method to look at the tasks and these tasks are represented as nodes in the graph. The edges between the tasks show their relation to one another. The features of the nodes can be the coordinates, arrival time, or other such relevant information and the edges between two nodes can show the distance between them. An important feature of this model is the ST-Graph Encoder. The Spatial-Correlation Encoding component uses a Graph Convolutional Network (GCN) to update node and edge embeddings based on their interactions. The Temporal Correlations Encoding component uses a Grated Recurrent Unit (GRU) to model the evolution of decision contexts over time. Grpah2Route uses a Graph-Based Personalised Route Decoder to predict future routes. This Decoder uses the worker-specific information and the graph structure to predict routes. An attention mechanism looks at the relevant parts of the graph during the decoding process and then a mask mechanism is used to filter the infeasible nodes which saves memory and improves prediction accuracy.

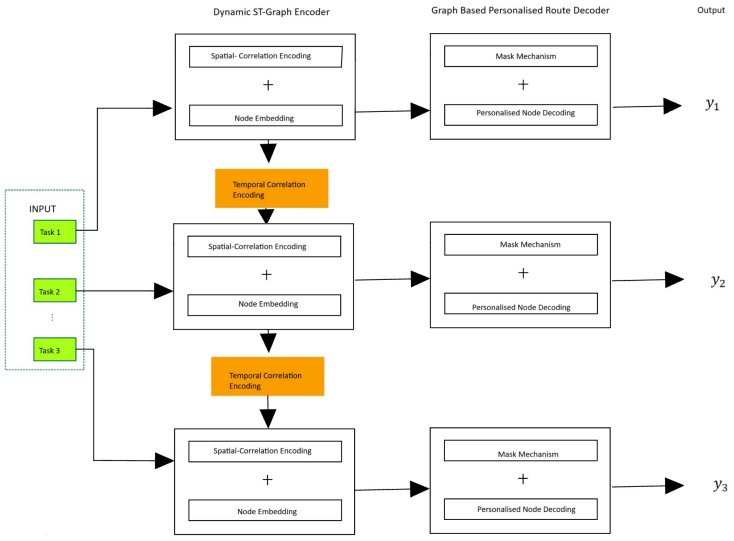

Figure 4.1: Graph2Route architecture

Figure 4.1 shows the architecture of the Graph2Route model which uses a Dynamic Spatio-Temporal(ST) Graph Encoder. This model processes tasks using a series of steps which start with the spatial correlation encoding and node embedding to capture spatial relationships of the tugboats represented in the data. The output is further processed by temporal correlation encoding to understand the temporal dependencies

depicted in the data. This encoded data is fed into a graph-based personalised route decoder that uses a mask mechanism to only focus on relevant nodes and a personalised node decoding process to generate the final route predictions. This approach allows for this route prediction model to adapt to different input scenarios and personalized the route predictions that reflect the individual factors of the specific tugboats.

4.5.2. Mathematical Model

The Graph2Route model uses a GCN for node and edge embeddings, GRU for temporal encoding, and an attention-based decoder for routing decisions. The mathematical model involves matrix multiplications, non-linear activations, softmax operations, and LSTM updates.
The future service routes of the tugboat is represented by Equation 4.19 where π_i represents the i-th node in the predicted route.

$$F_C(G_{wt}) = \pi_1, \pi_2, \ldots, \pi_{|V_{Ut}|} \tag{4.19}$$

The Dynamic ST-Graph Encoder uses the node and edge embeddings at each time step using two encoding modules:

- Spatial-Correlation Encoding: A GCN is used to capture the spatial-temporal correlations between the different nodes using both node and edge features to learn more about the system. It updates d_h-dimensional node and edge embeddings across L layers. Let $h_i^{(l)}$ be the embedding of node i at layer l, and $z_{ij}^{(l)}$ be the edge embedding at layer l. Initial embeddings are $h_i^0 = x_i$ and $z_{ij}^0 = e_{ij}$.

$$h_i^{(l+1)} = f\left(h_i^{(l)}, \mathsf{Agg}\left(\{h_j^{(l)}, z_{ij}^{(l)} : j \in N_i\}\right)\right) \tag{4.20}$$

$$z_{ij}^{(l+1)} = g\left(z_{ij}^{(l)}, \mathsf{Agg}\left(\{h_i^{(l)}, h_j^{(l)}\}\right)\right) \tag{4.21}$$

  N_i is the set of neighbors of node i, Agg is the aggregation function, and f and g are defined by:

$$h_i^{(l+1)} = h_i^{(l)} + \sigma(\mathsf{BN}(W_1^{(l)} h_i^{(l)} + \sum_{j \in N_i} \eta_{ij}^{(l)} \odot W_2^{(l)} h_j^{(l)}))$$

$$z_{ij}^{(l+1)} = z_{ij}^{(l)} + \sigma(\mathsf{BN}(W_3^{(l)} z_{ij}^{(l)} + W_4^{(l)} h_i^{(l)} + W_5^{(l)} h_j^{(l)}))$$

- Temporal Correlation Ecnoding: An RNN studies the decision context and uses the historical information to embed these features for the decoder. At each time step t, the encoder takes the graph G_{wt} and previous node embeddings H_{t-1} as input, producing updated embeddings H_t.

$$H_t = \mathsf{GRU}(G_{wt}, H_{t-1}) \tag{4.22}$$

The Decoding process uses the chain rule to find conditional probabilities of the possible actions it can take and then chooses the action with the highest probability as shown in Equation 4.23 [9], where s is the problem instance, and θ represents all trainable parameters. $f(s, \theta_e)$ is the dynamic graph encoder, and θ_d is the trainable parameter of the decoder.

$$p(\pi \mid s;\theta) = \prod_{j=1}^{n} p(\pi_j \mid s, \pi_{1:j-1}, w;\theta) = \prod_{j=1}^{n} p(\pi_j \mid f(s,\theta_e), \pi_{1:j-1}, w;\theta_d) \tag{4.23}$$

The decoding process generates the probabilities of the feasible locations that the tugboat can visit next. This can be seen as a multi-class classification problem and the cross-entropy loss function is used to calculate the loss of the model in Equation 4.24 where W is the set of workers, T is the set of the total sampling time steps, y_i is the order of task i in the label of that sample, and $p(y_i \mid \cdot)$ is the predicted probability of task i predicted by the model.

$$L = -\sum_{w \in W} \sum_{t \in T} \sum_{i \in \pi_t : t'} y_i \log(p(y_i \mid \theta)) \tag{4.24}$$

Graph2route also uses an Osquare algorithm that predicts the routes by initialising an empty route π and padding the already outputted locations to avoid repeating locations. Then the features of the locations are concatenated and LightGBM is used for the prediction. The task with the highest score that has not yet been outputted is selected as the next location and is then appended to the route. A mask mechanism is used to avoid predictions not satisfying the system's constraints.

4.5.3. Advantages of Graph2Route

Graph2Route improves the accuracy of predictions as it is able to fully encode the spacio-temporal correlations between tasks. The model uses GCNs and GRUs that allow it to look at current and historical decision decisions.
This method also reduces the search space needed using the mask mechanism. This feature excludes the infeasible nodes during decoding, filters out unreasonable routes and improves prediction quality. This exclusion feature ensures that the predictions are more accurate.
Graph2Route also adapts to the newest tasks by updating the graph and embeddings. This is very useful in real-world scenarios where tasks are added continuously and the predictions are changed dynamically. These characteristics of Graph2Route make it a very powerful method for route prediction.

4.6. Greedy Methods

The Greedy methods can be split into two different methods. Time Greedy and Distance Greedy.

4.6.1. Time Greedy Method

This method aims to minimise the time required to complete all the tasks assigned to a tugboat. It selects the route with the shortest time between nodes.

The key components of this method are:

- Initialisation:
  T is defined as the time vector, and T_j represents the time to reach node j. **M** is defined as the mask vector. This is used to track the visited nodes. ***P*** is defined as the path vector which stores the list of the visited nodes.

- Iterations:
  At each time step ***t***:

$$j = \arg\min_k \{T_k \mid M_k = 0\} \tag{4.25}$$

  Mark node ***j*** as visited and added to the path. $M_j = 1$ and $P_t = j$ and update current node to ***j***.
- Termination:
  The process repeats until all nodes are visited.

4.6.2. Distance Greedy Method

The distance greedy method is used to find out the shortest path for the solution in terms of distance. The key components of this method are:

- Initialisation:
  D is defined as the distance matrix where D_ij is the distance between noides ***i*** and ***j***. ***M*** and ***P*** are the mask vector and path vectors respectively. ***M*** tracks the visited nodes and ***P*** stores the order of the visited nodes.
- Iterations:
  At each time step ***t***: Select the unvisited node ***j*** that minimises the distance from current node ***i***.

$$j = \arg\min_k \{D_{ik} \mid M_k = 0\} \tag{4.26}$$

  Mark node ***j*** as visited and added to the path. $M_j = 1$ and $P_t = j$ and update the current node to ***j***.
- Termination:
  The process repeats until all nodes are visited.

These models are a simple route optimisation method by making locally optimal choices at each step on the time or distance metrics.

4.7. DRL4Route

Deep Reinforcement Learning (DRL) is an advanced machine learning method that uses both Reinforcement Learning (RL) and Deep Learning. The agents' objective in this method is to learn the optimal actions for the Route Prediction problem using a trial and error system. The model DRL4Route [25] uses the deep learning models as agents and the test criteria as the reward in reinforcement learning.

Reinforcement-Guided Route Prediction

Route Prediction can be seen as a step-by-step learning process where each step on the route influences future decisions. A discrete finite-horizon Markov Decision Process (MDP) [2] can be used to represent this decision-making process shown in Equation 4.27, where S is the set of states, A is the set of actions, $P : S \times A \times S \to \mathbb{R}^+$ is the transition probability, $R : S \times A \to \mathbb{R}$ is the reward function, $s_0 : S \to \mathbb{R}^+$ is the initial state distribution, $\gamma \in [0, 1]$ is a discount factor, and T is the total time steps determined by the number of unfinished tasks.

$$M = (S, A, P, R, s_0, \gamma, T) \tag{4.27}$$

The action that a route prediction agent makes is shown as θ. This action has a reward r_t associated with it based on how it performs under the evaluation metric being used. The objective of this training is to maximise the total reward of the tasks which is seen in Equation 4.28. γ is the discount factor that balances the importance of immediate vs future rewards.

$$\theta^* = \arg\max_{\theta} \mathbb{E}_{\pi_\theta} \left[\sum_{t=1}^{T} \gamma^t r_t \right] \tag{4.28}$$

At this stage, the route prediction agent uses an encoder-decoder architecture. The encoder is used to compute the representations *E* for unfinished tasks and the Decoder predicts the route $\hat{Y}$ step-by-step based on the embeddings, *E* and previous outputs as shown in Equation 4.30. h_t is the hidden state of the decoder, o_t is the output probability distribution at the t-th decoding step, and *C* is the route constraints.

$$E = \text{Encoder}(O_w) \tag{4.29}$$

$$o_t, h_{t+1} = \text{DecoderCell}(E, C, h_t, \hat{Y}_{1:t-1}) \tag{4.30}$$

The state of the system shows the condition of the system at the t-th decoding step of an agent which considers all the factors that make up the system and all the information an agent of the system has to make the decisions at each time step as shown in Equation 4.31

$$s_t = (E, C, h_t, \hat{Y}_{1:t-1}) \tag{4.31}$$

An action involves selecting a task $\hat{y}_t$ from the current unfinished tasks based on the current state of the system. A set of actions $(a_1, \ldots, a_n \in A = A_1 \text{ x } \ldots \text{ x } A_n)$. Based on the actions taken by the agent, a reward is given to the agent based on the evaluation metric used to judge the action which can be seen in chapter 5. The goal of DRL is to gain the maximum cumulative reward at the end of the route prediction.

The system described above evaluates the state of the system at every time step and determines an action from the policy π_θ. Depending on the output of the agent's action at time 't', it receives a reward 'r_t'.

$$L_{\theta_a} = -\mathbb{E}_{\pi_{\theta_a}} \left[r(\hat{y}_1, \cdots, \hat{y}_n) \right] \tag{4.32}$$

where $\hat{y}_1, \cdots, \hat{y}_n \sim \pi_{\theta_a}$, and they are sampled from the policy π_{θ_a}. $r(\hat{y}_1, \cdots, \hat{y}_n)$ is the reward associated with the predicted route $\hat{y}_1, \cdots, \hat{y}_n$. Equation 4.32 along with the N sampled routes from the agent's policy π_θ becomes the following function for loss $L_{\theta a}$.

$$\nabla_{\theta_a} L_{\theta_a} = -\frac{1}{N} \sum_{i=1}^{N} \left[\nabla_{\theta_a} \log \pi_{\theta_a}(\hat{y}_{i,1}, \cdots, \hat{y}_{i,n}) r(\hat{y}_{i,1}, \cdots, \hat{y}_{i,n}) \right] \tag{4.33}$$

Equation 4.33 is based on an algorithm called Reinforce from [46]. However, this model would have to run its full course of action to observe the reward, which could

lead to error accumulation.

To prevent this particular issue, an Actor-Critic architecture [37] is set up which reduces the variance of the policy gradient estimates by providing feedback at every time step. The Actor (agent) is in charge of making the actions at every time interval based on the actions available to it. The Critic evaluates these decisions based on evaluation metrics to provide a reward at every time step based on the state-value function. The Actor, which is the route prediction agent, learns from the Critic's feedback and updates its policy distributions toward the direction that the Critic suggests.

The policy π_{θ_a} defines the values of the state-action pair $Q(s_t, a_t)$ and the value $V(s_t)$ in Equation 4.34 and Equation 4.35 and the state-action value function can be computed recursively with dynamic programming as per Equation 4.36

$$Q_{\pi_{\theta_a}}(s_t, a_t) = \mathbb{E}_{\pi_{\theta_a}}\left[r(\hat{y}_t, \cdots, \hat{y}_n) \mid s = s_t, a = a_t\right] \tag{4.34}$$

$$V_{\pi_{\theta_a}}(s_t) = \mathbb{E}_{a_t \sim \pi_{\theta_a}}\left[Q_{\pi_{\theta_a}}(s_t, a = a_t)\right] \tag{4.35}$$

$$Q_{\pi_{\theta_a}}(s_t, a_t) = \mathbb{E}_{s_{t+1}}\left[r_t + \gamma \mathbb{E}_{a_{t+1} \sim \pi_{\theta_a}(s_{t+1})}\left[Q^{\pi_{\theta_a}}(s_{t+1}, a_{t+1})\right]\right] \tag{4.36}$$

The Critic shares the parameters of the system with the Actor to make use of the spatio-temporal information of unfinished tasks and estimate the state function. The predicted value of the state function is then used to estimate the advantage function as per Equation 4.37

$$\begin{aligned} A^{\pi_{\theta_a}}(s_t, a_t) &= Q_{\pi_{\theta_a}}(s_t, a_t) - V_{\pi_{\theta_a}}(s_t) \\ &= r_t + \gamma \mathbb{E}_{s_{t+1} \sim \pi_{\theta_a}(s_{t+1} \mid s_t)}\left[V_{\pi_{\theta_a}}(s_{t+1})\right] - V_{\pi_{\theta_a}}(s_t) \end{aligned} \tag{4.37}$$

$$A_{\pi_{\theta_a}}(s_t, a_t) \approx r_t + \gamma V_{\pi_{\theta_a}}(s_{t+1}) - V^{\pi_{\theta_a}}(s_t) \tag{4.38}$$

Equation 4.37 gives the ability to sample an unfinished task set and estimate the advantage function to by calculating the expectation value function in state s_{t+1}, which is shown in Equation 4.38.
The value function *V* checks the quality of the policy in a specific state s_t. The *Q* checks the value of choosing an action when in that state. The advantage function measures the difference between the *Q*-function and the value function *V* to check the relative benefit of each action.
Multiple unfinished task sets are sampled for the advantage function to minimise the variance of the gradient estimates. The loss in the Actor-Critic setup is measured in Equation 4.39.

$$L_{\theta_a} = \frac{1}{N} \sum_{i=1}^{N} \sum_{t \in T} \log \pi_{\theta_a}(a_{i,t} \mid s_{i,t}) A_{\pi_{\theta_a}}(s_{i,t}, a_{i,t}) \tag{4.39}$$

As stated earlier,π_{θ_a} is the policy function modeled by the actor network. The critic tries to estimate $r(\hat{y}_t, \cdots, \hat{y}_n)$ for the model at each decoding step t. The predicted value $b_{\theta_c, \theta_a}(s_t)$ of the critic is called the "baseline". The critic is trained by considering it as a regression problem, and the value function is trained using a loss function calculation from [11] which is less sensitive to outliers than L_2 loss as depicted in Equation 4.40. $smoothL_1$ is given by Equation 4.41

$$L_{\theta_c} = \frac{1}{N}\sum_{i=1}^{N}\sum_{t\in T}\text{smooth}L_1(b_{\theta_c,\theta_a}(s_{i,t}) - r(\hat{y}_{i,t},\cdots,\hat{y}_{i,n})) \tag{4.40}$$

$$\text{smooth}L_1(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases} \tag{4.41}$$

The system is first pre-trained to optimise the parameters of the route-prediction agent by using a maximum-likelihood objective and minimising the cross-entropy loss using Equation 4.42

$$L_{CE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{t\in T}\log(P(y_{i,t} \mid \theta_a)) \tag{4.42}$$

The next phase is the joint training of the actor and the critic by using Equation 4.43 where $\theta_{ac} = \theta_a \cup \theta_c$ represents the parameters of both the actor and the critic. α_{θ_c}, α_{θ_a}, and α_{CE} are hyper-parameters to control the weight of different loss functions.

$$L_{\theta_{ac}} = \alpha_{\theta_c}L_{\theta_c} + \alpha_{\theta_a}L_{\theta_a} + \alpha_{CE}L_{CE} \tag{4.43}$$

The prediction step involves the actor taking the unfinished task set as an input and predicting the whole route step by step. The task to select at each decoding step is given by Equation 4.44 where $p_{t,k}$ is the output probability of task k at the t-th step.

$$\hat{y}_t = \arg\max_k p_{t,k} \tag{4.44}$$

4.7.1. Generalised Advantage Method

Generalised Advantage Estimation is a method used to reduce the biasing due present in the actor-critic model. Actor-critic models usually have low variance due to the batch training method used and the baseline reward feature of the critic. As these models usually use neural networks to approximate the *V* function and use this value to generate the va;ue of the *Q* function, the value of the *Q* function is biased if the system doesn't accurately approximate the *V* function. Models based on REINFORCE have low bias but can have high variance as the expected discounted reward is generated using sampling. To find a trade-off between the bias and variance in estimating the expected gradient of the policy-based loss function, the generalised advantage estimation method [34] is used. This is shown by Equation 4.45 which is an edited version of Equation 4.37. λ is used to control the trade-off between the bias and variance. Larger values of λ lead to high variance and low bias and vice-versa.

$$A^{\text{GAE}(\gamma,\lambda)}_{\pi_{\theta_a}}(s_t,a_t) = \sum_{t'=t}^{T}(\gamma\lambda)^{(t'-t)}\left(r(s_{t'},a_{t'}) + \gamma V_{\pi_{\theta_a}}(s_{t'+1}) - V_{\pi_{\theta_a}}(s_{t'})\right) \tag{4.45}$$

Actor

The actor plays the role of the route prediction agent in this system. The Actor has an encoder-decoder architecture, as described in section 4.2. The encoder uses the

spatio-temporal features of the unfinished tasks to generate their representations. The Decoder layer selects a task $\hat{y}_t$ at time instant *t* from the unfinished task set and this task is used as the input in the following input step. This process is repeated until all the tasks in the unfinished task set are completed.

According to the chain rule, the probability of an output service route $\hat{Y}_w$ is expressed as a product of conditional probabilities in Equation 4.46

$$P(\hat{Y}_w \mid O_w; \theta_a) = \prod_{t=1}^{n} P(\hat{y}_t \mid O_w, C, \hat{Y}_{1:t-1}; \theta_a) \tag{4.46}$$

The Encoder used is a transformer encoder, which consists of an *N* number of transformer blocks to integrate the spatio-temporal features of the unfinished tasks. Each transformer block is made up of a multi-head attention (MHA) layer and a feed-forward network (FFN) layer. The embeddings from these blocks are shown as $l \in \{1, \cdots, K\}$ as e_j^l and n_{head} is the number of heads.

$$\hat{e}_j = \text{BN}_l \left(e_j^{l-1} + \text{MHA}_l \left(\text{head}_1^{l-1}, \cdots, \text{head}_{n\text{head}}^{l-1} \right) \right) \tag{4.47}$$

$$e_j^l = \text{BN}_l \left(\hat{e}_j + \text{FFN}(\hat{e}_j) \right) \tag{4.48}$$

The Decoder layer uses an attention-based recurrent layer to generate the prediction route based on the encodings made by the encoder at time *t* by outputting task $\hat{y}_t$. It also uses the knowledge of the system, that is, the previous outputs of the decoder before time step *t*. The decoder uses LSTM along with an attention mechanism to understand the probability shown in Equation 4.46. The initial input to the decoder is all of the task embeddings $\{e_1, e_2, \cdots, e_n\}$, the aggregated embedding $e = \frac{1}{n} \sum_{j=1}^{n} e_j$, and the randomly initialised hidden state of the LSTM h_0. The next step is to generate the attention score for all the unfinished tasks at each decoding step *t*, and masks ($u_j^t = -\infty$) for the tasks which have been output before the time step to meet the route constraints C.

$$u_j^t = \begin{cases} v^T \tanh(W_1 e_j + W_2 h_t) & \text{if } j \neq \hat{y}_{t'}, \forall t' \leq t \\ -\infty & \text{otherwise} \end{cases} \tag{4.49}$$

where $W_1, W_2 \in \mathbb{R}^{d_h \times d_h}$ and $v \in \mathbb{R}^{d_h}$ are learnable parameters. Finally, the output probability p_j^t of task j at step t by a softmax layer is shown in

Critic Network

In policy-based actor-critic networks, the critic provides an estimate of the state-value function to reduce the variance of the gradient estimation. The critic is therefore expected to understand all the information about the environment of the system. This environment includes the spatio-temporal information of the unfinished tasks, route constraints, tasks that have been output already in the decoding process. Since the output probability distribution at each decoding step is determined by the state and has sufficient information to evaluate the state-value function, the parameters of the actor are shared with the critic. The output probability distribution at each decoding step is then input into the critic. In practice, the critic, which is parameterised by θ_c

is implemented as a feed-forward network functioning as a regression layer and is trained simultaneously with the actor network.

5

Evaulation Metrics

A number of evaluation metrics are used to understand the performance of the route predictions performed by the various methods mentioned in chapter 4.

5.1. Kendall Rank Correlation

Kendall Rank Correlation (KRC) [1] is a metric used to evaluate the ordinal association between two sequences. This shows how the two sets of data match with each other. KRC checks which ranked items of one list match the order in which they are ranked in the other.
This matching is determined by observing pairs of items from both sequences and checking if their relative ranking in their sequence is agreeable. A pair of items ***(i,j)*** is considered concordant if both sequences rank these items in the same order. This means that is item ***i*** is ranked higher than item ***j*** in both sequences, or vice versa, the pair is concordant. Else, the pair is call discordant.
When calculating KRC, the nodes in the predicted sequences are divided into two sets. Set V_in consists of nodes that are in the predicted and actual sequence. V_not consists of nodes that are only in the predicted sequence. The order of the items in V_in is known but the order in V_not is not. It is assumed that the items in V_in precede the ones in V_not.

The following is the definition and the steps in the calculation of KRC.
For sequences $\hat{Y}_w$ and Y_w:

A node pair (i, j) is concordant if:

$$(R_{\hat{Y}}(i) > R_{\hat{Y}}(j) \text{ and } R_Y(i) > R_Y(j)) \text{ or } (R_{\hat{Y}}(i) < R_{\hat{Y}}(j) \text{ and } R_Y(i) < R_Y(j)) \quad (5.1)$$

To calculate KRC, nodes in the prediction are first divided into two sets as shown in Equation 5.2 and Equation 5.3. The node pairs to be compares are determined in Equation 5.4. N_c and N_d are the number of concordant and discordant pairs respectively. KRC is calculates using the formula shown in Equation 5.5.

$$V_{\text{in}} = \{\hat{y}_i \mid \hat{y}_i \in Y_w\} \quad (5.2)$$

$$V_{\text{not}} = \{\hat{y}_i \mid \hat{y}_i \notin Y_w\} \quad (5.3)$$

$$(i,j) \mid i,j \in V_{\text{in}} \text{ and } i \neq j\} \cup \{(i,j) \mid i \in V_{\text{in}} \text{ and } j \in V_{\text{not}}\} \tag{5.4}$$

$$\text{KRC} = \frac{N_c - N_d}{N_c + N_d} \tag{5.5}$$

The larger the value of KRC, the higher the correlation between the predicted route and the actual route.

5.2. Edit Distance

Edit distance is a metric used to show the difference between two sequences by calculating the minimum number of operations required to transform one sequence into the other [27]. The operations considered are insertion, deletion, and substituion of characters. The most common form of edit distance is the Levenshtein distance. This distance is defined as the minimum number of single character edits needed to change one stringinto the other.
The calculation of Levenshtein distance between two strings requires a few steps:

- A matrix of size ***(length of string A + 1)*** by ***(length of string B + 1)*** is initialised. The first row and first column of this matrix are initialised with index values.
- The matrix is then filled by iterating through each cell and computing the cost of converting the substrings of the given strings. If the characters being compared are the same, the cost remains the same as
- The matrix is then filled by iterating through each cell and computing the cost of converting the substrings of the given strings. If the characters being compared are the same, the cost remains the same as that of converting the previous substrings. A minimum cost is considered if the characters differ. These are: insertion (adding a character to one string), deletion (removing a character from one string), and substitution (replacing one character with another).
- The cost for the current cell is the minimum of these operations plus one. Once the matric is full, the value in the bottom right cell is the Levenshtein distance between two strings.

5.3. Location Square Deviation and Location Mean Deviation

The Location Square Deviation (LSD) and the Location Mean Deviation (LMD) measure the degree to which predictions deviate from the actual labels. The values in both these metrics are inversely proportional to the accuracy of the prediction.

Location Square Deviation (LSD)

LSD measures the average squared difference between the predicted values and the actual values which can be seen in Equation 5.6

$$\text{LSD} = \frac{1}{m} \sum_{i=1}^{m} \left(R_Y(y_i) - R_Y(\hat{y}_i)\right)^2 \tag{5.6}$$

where:

- m is the number of observations,
- y_i is the actual value for the i-th observation,
- $\hat{y}_i$ is the predicted value for the i-th observation,
- $R_Y(y_i)$ is the rank of the actual value y_i,
- $R_Y(\hat{y}_i)$ is the rank of the predicted value $\hat{y}_i$.

Large deviations in the metric are penalised more because the differences are squared. It is useful to find significant prediction errors.

5.3.1. Location Mean Deviation (LMD)

The LMD measures the average absolute difference between the predicted and actual values and can be seen in Equation 5.7

$$\text{LMD} = \frac{1}{m}\sum_{i=1}^{m}|R_Y(y_i) - R_Y(\hat{y}_i)| \tag{5.7}$$

where:

- m is the number of observations,
- y_i is the actual value (label) for the i-th observation,
- $\hat{y}_i$ is the predicted value for the i-th observation,
- $R_Y(y_i)$ is the rank of the actual value y_i,
- $R_Y(\hat{y}_i)$ is the rank of the predicted value $\hat{y}_i$.

The LMD metric shows a straightforward average of deviations.

LSD is good at identifying large deviations due to the squaring of differences and LMD gives the overall average of the deviations by comparing the average absolute difference between the predicted and actual values.

5.4. Hit rate @k

This metric measures how many of the top-k predicted items are also in the top-k items of the actual labels. It provides a sense of how well the top-k predictions match the true top-k items which shows the accuracy of the model in ranking the most relevant items as seen in [15] and [25]. The HR@k formulation can be seen in Equation 5.8 where $\hat{Y}[1:k]$ represents the top-k predicted items and $Y[1:k]$ represents the top-k actual items. The numerator indicates the number of common items between the predicted and actual top-k lists, and the denominator normalises this count by k.

$$\text{HR@k} = \frac{|\hat{Y}[1:k] \cap Y[1:k]|}{k} \tag{5.8}$$

5.5. Accuracy @k

Accuracy@k (ACC@k) is a metric similar to HR@k but it is sticker than HR@k. It checks whether the exact order of the top-k predicted items matches the order of the top-k actual items. This metric is more stringent because it requires not only the

presence of the correct items but also their correct ordering. ACC@k is defined in Equation 5.9 where $I(\cdot)$ is an indicator function that equals 1 if the predicted item $\hat{y}_i$ matches the actual item y_i at position i, and 0 otherwise. This formula checks that ACC@k is 1 only if all top-k predicted items exactly match the corresponding top-k actual items in both presence and order, otherwise, it is 0.

$$\text{ACC@k} = \prod_{i=1}^{k} I(\hat{y}_i, y_i) \tag{5.9}$$

5.6. Dist

Dist is a metric that measures the average distance between corresponding points in the predicted and actual routes. This is an additional metric to measure the effectiveness of the route prediction by showcasing the closeness of the predicted sequence with that of the actual sequence. To do this, the metric uses a haversine function that measures the haversine distance between the two distances and saves this distance. In the end, all of these distances are added up and divided by the total number of time the metric function is called to generate the average deviation between the predicted and actual routes in kilometres. The lower the value of this value, the better the functioning of the route prediction method. The metric Dist is used to check the average distance between the points on the predicted route and the actual route. The goal of the route prediction is to get as small a value as possible for this particular metric with the overall goal for this dist to be 0, which would indicate a perfect prediction.

The metrics in chapter 5, namely, KRC, ED, LSD, and LMD are used to compare the similarity of the predicted and actual routes. Metrics such as HR@k and ACC@k check the similarity from a local perspective. Higher values of KRC, HR@k, ACC@k, and lower values of LSD,LMD, and ED indicate a better performance of the route prediction method.

6

Results

Tables for eval_max = 11 and 25

	krc	ed	lsd	lmd	hr@1	acc@3	method
	0.5480	1.5019	2.7746	1.0321	0.5233	0.2903	deeproute
	0.4156	1.5798	4.1757	1.3321	0.4167	0.2302	distance_greedy
	0.3848	1.5860	3.5307	1.2321	0.4744	0.2140	fdnet
eval_max = 11	0.5301	1.5191	2.8336	1.0562	0.5088	0.2763	graph2route
	0.3862	1.8043	4.4171	1.5276	0.3043	0.1273	osqure
	0.4447	1.7147	3.9145	1.3299	0.3687	0.1976	time_greedy
	0.5490	1.4868	2.5945	1.0017	0.5213	0.2941	DRL4Route_REINFORCE
	0.5503	1.4941	2.5730	1.0030	0.5215	0.2961	DRL4Route_REINFORCE_GAE

	krc	ed	lsd	lmd	hr@1	acc@3	method
	0.5476	1.5189	2.8012	1.0361	0.5230	0.2899	deeproute
	0.4150	1.5982	4.2158	1.3372	0.4164	0.2299	distance_greedy
	0.3848	1.5860	3.5307	1.2321	0.4744	0.2140	fdnet
eval_max = 25	0.5297	1.5364	2.8600	1.0602	0.5086	0.2758	graph2route
	0.3848	1.8297	4.4850	1.5353	0.3034	0.1269	osqure
	0.4441	1.7338	3.9473	1.3345	0.3681	0.1972	time_greedy
	0.5485	1.5043	2.6226	1.0072	0.5209	0.2935	DRL4Route_REINFORCE
	0.5498	1.5115	2.6013	1.0059	0.5210	0.2955	DRL4Route_REINFORCE_GAE

Table 6.1: Comparison of Different Methods for eval_max 11 and 25

Table 6.1 shows the results of the different route prediction methods from chapter 4 evaluated using the metrics mentioned in chapter 5. From this table, it can be observed that the baseline methods, Distance-Greedy and Time-Greedy methods ,perform the worst. These methods show low KRC, and hit rates, and high ED, and LSD values. This is because these methods focus only on the time or distance reduction factors and aim to reduce distance and time of the next possible step respectively. This does not focus on the overall goal and therefore performs poorly when compared to the other methods. Osquare, with its XGBoost method, also performs quite poorly compared to the deep learning methods because they use more sophisticated encoding and decoding processes and are better at handling sequential data, identifying patterns and extracting relevant features. In the deep learning methods, Graph2Route performs better than FDNet because it uses a dynamic spatio-temporal graph encoder and graph-based personalised route decoder that enable it to understand the input data better and make more accurate route predictions and score highly in the KRC, hit rate and ACC metrics and also have low ed, lsd and lmd scores which indicate the superiority of the Graph2route method and indicate its effectiveness in route prediction problems. Deeproute and Graph2Route perform very similarly but Deeproute edges

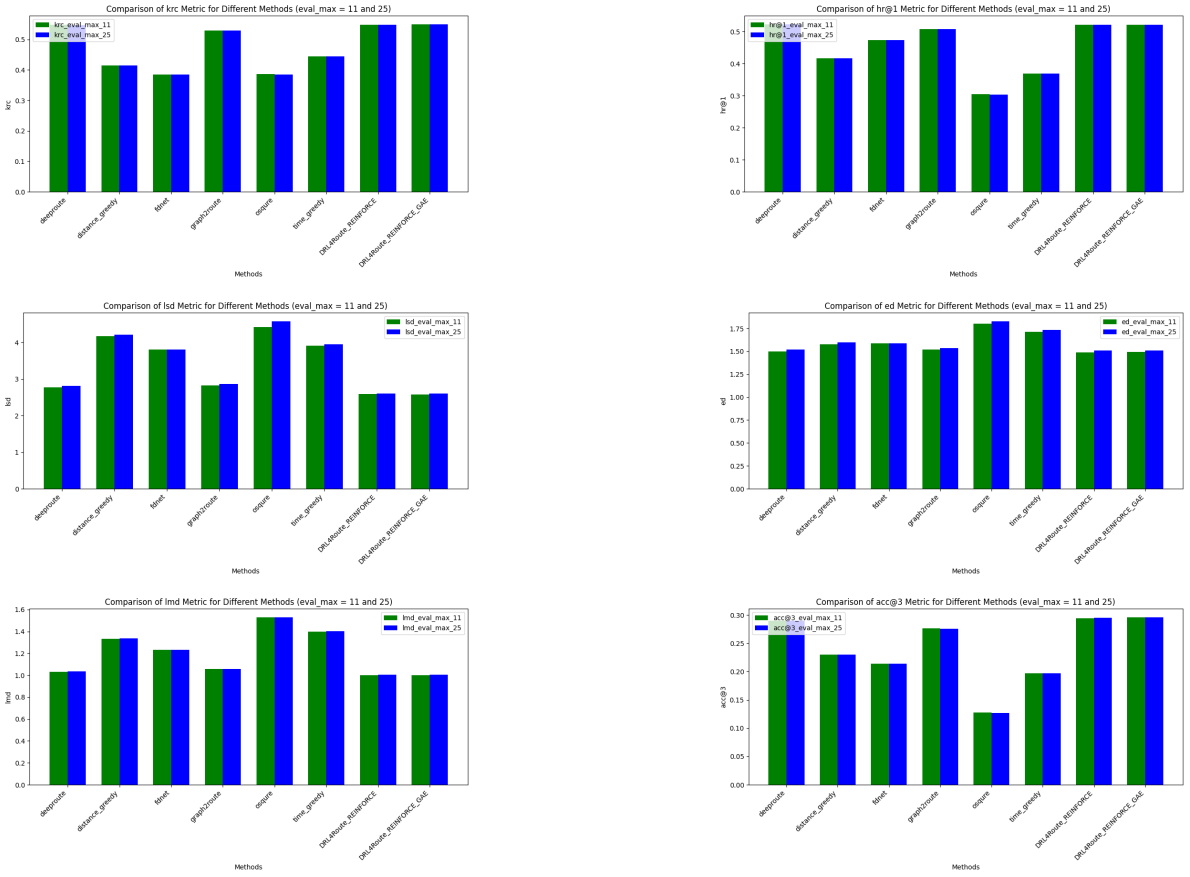

Figure 6.1: fig: Performance of the route prediction methods

out in front because of its powerful transformer encoder which dynamically adjusts it to factors such as distance and time and the attention mechanism-based decoders determine the relevant part of the data to make accurate predictions which are seen particularly in the KRC and hit rates which show that a higher percentage of deep-route's predictions align with the actual route that is provided .
DRL4Route-REINFORCE improves on the above methods because this method uses the evaluation metrics mentioned in chapter 5 to optimise the solution directly. This means the solution solves the issues faced by cross-entropy methods with different goals during the training and testing phases. Combining deep learning and reinforcement learning and optimisation using the evaluation metrics allows DRL4Route to have a more accurate route prediction model which is reflected by the metric values shown in Table 6.1. DRL4Route-REINFORCE-GAE uses the Generalised Advantage Estimator which balances the bias and variance of the policy gradient. By balancing the bias and variance, this method is able to make the most accurate predictions which can be inferred by the high KRC, Hit rate and ACC values and the low ed, lsd, lmd values.

The above Table 6.2 compares the average distance between corresponding nodes in the predicted route and the actual route. This data is also visualised in Figure 6.2. Based on the evaluation metrics and the results discussed in Table 6.1, DRL4Route and DRL4Route-GAE emerged as the best methods for route prediction from the methods that were described in chapter 4 when applied to the same dataset. These two

Method	Incoming harbour EURO	Outgoing harbour EURO	Incoming berth EUROPAH
DRL4Route	1.2091	1.7904	1.9717
DRL4RouteGAE	1.2090	1.79	1.9716
Inverse optimisation	1.1780	1.7860	1.4500
Benchmark - kotug	1.1020	2.1760	1.5170

Table 6.2: Comparison of different methods based on the average distance between points in the predicted vs actual route

methods were selected for further analysis using the average distance metric, ”dist”, as discussed in chapter 5. The other methods used to compare this metric's results are an inverse optimisation model and a benchmarking model developed by KOTUG. For this evaluation, data was drawn from the incoming and outgoing vessels at the EURO harbour and the incoming vessels to the EUROPAH berth. The large dataset was split to focus specifically on these routes. The results indicate that while the two DRL methods performed well overall, they did not outperform the inverse optimisation model. The inverse optimisation model achieved a marginally lower average in the incoming and outgoing EURO harbour segments and a significant margin in the incoming EUROPAH berth segment. However, The two DRL models performed better overall than the Benchmarking method employed by KOTUG. This fluctuation between the four methods could be due to the difference in the datasets and preprocessing methods used to prepare the data. For a more concrete understanding of the accuracy of the models, the inverse optimisation and KOTUG models should be evaluated using the other metrics discussed in chapter 5.

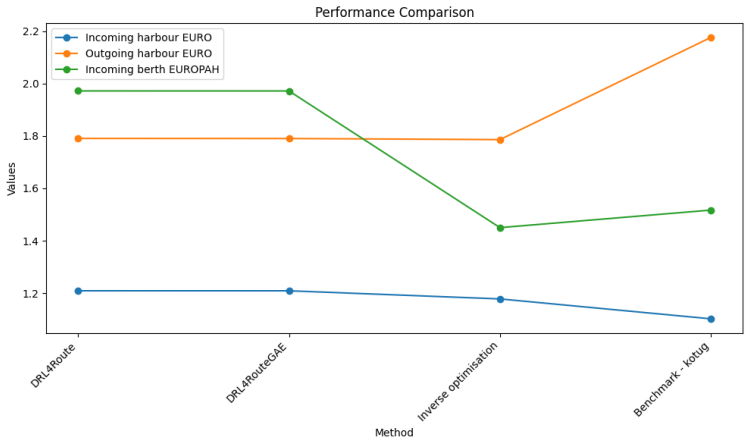

Figure 6.2: Average distance between predicted and actual nodes

DRL4Route and DRL4Route-GAE have also been used on other route prediction problems. [25] talks about the application of these models for the pick-up and delivery route prediction scenarios based on extensive real-world datasets in multiple cities in China. A notable example is the performance of these models in the city oh Hangzhou, which can be seen in Table 6.3 and Figure 6.3.
The results presented in Table 6.3 clearly showcase the superior performance of the DRL4Route and DRL4Route-GAE models when compared to both traditional heuristic methods, such as the Time and Distance-Greedy methods, and other deep learning-based approaches (e.g., FDNET, DeepRoute). DRL4Route and DRL4Route-GAE models achieve higher Kendall's Rank Correlation (KRC), Hit Rate (HR@1), and Ac-

krc	ed	lsd	lmd	hr@1	acc@3	method
0.4192	1.78	6.85	1.70	0.3315	0.2032	Time-Greedy
0.52.68	1.48	5.02	1.27	0.5168	0.3413	Distance-Greedy
0.5547	1.50	4.14	1.18	0.5276	0.3322	FDNET
0.5861	1.45	3.71	1.10	0.5476	0.3464	DeepRoute
0.6063	1.43	3.47	1.05	0.5645	0.3612	Graph2Route
0.6057	1.05	3.47	1.05	0.5585	0.3574	DRL4Route-REINFORCE
0.6147	1.41	3.44	1.03	0.5772	0.3612	DRL4Route-GAE

Table 6.3: Performance comparison of various methods on different metrics.

curacy at Top-3 (ACC@3) values than the other methods in both Table 6.1 and Table 6.3 which indicate a higher degree of concordance between predicted and actual routes. This suggests that the DRL-based methods are significantly better at ranking the predicted routes more closely to the real-world outcomes. The HR@1 metric, which measures the likelihood of the top-predicted route being correct, is particularly high for DRL4Route-GAE and DRL4Route, significantly outperforming traditional heuristic methods such as Time-Greedy and Distance-Greedy. The low error values for Edit Distance (ED), Location Square Deviation (LSD), and Location Mean Deviation(LMD) metrics for DRL4Route and DRL4Route-GAE further confirm the ability of these models to provide precise and accurate route predictions with minimal deviation from the true paths. These results demonstrate the effectiveness of DRL4Route and DRL4Route-GAE models for complex route prediction tasks. Their ability to outperform traditional methods, as well as other deep learning models, suggests that DRL-based approaches are highly suitable for real-world applications in the domain of logistics, pick-up, and delivery services. The combination of high KRC, HR@1, and ACC@3 values, coupled with low ED, LSD, and LMD values, demonstrates that DRL4Route and DRL4Route-GAE are robust models capable of delivering highly accurate and reliable solutions to route prediction problems.

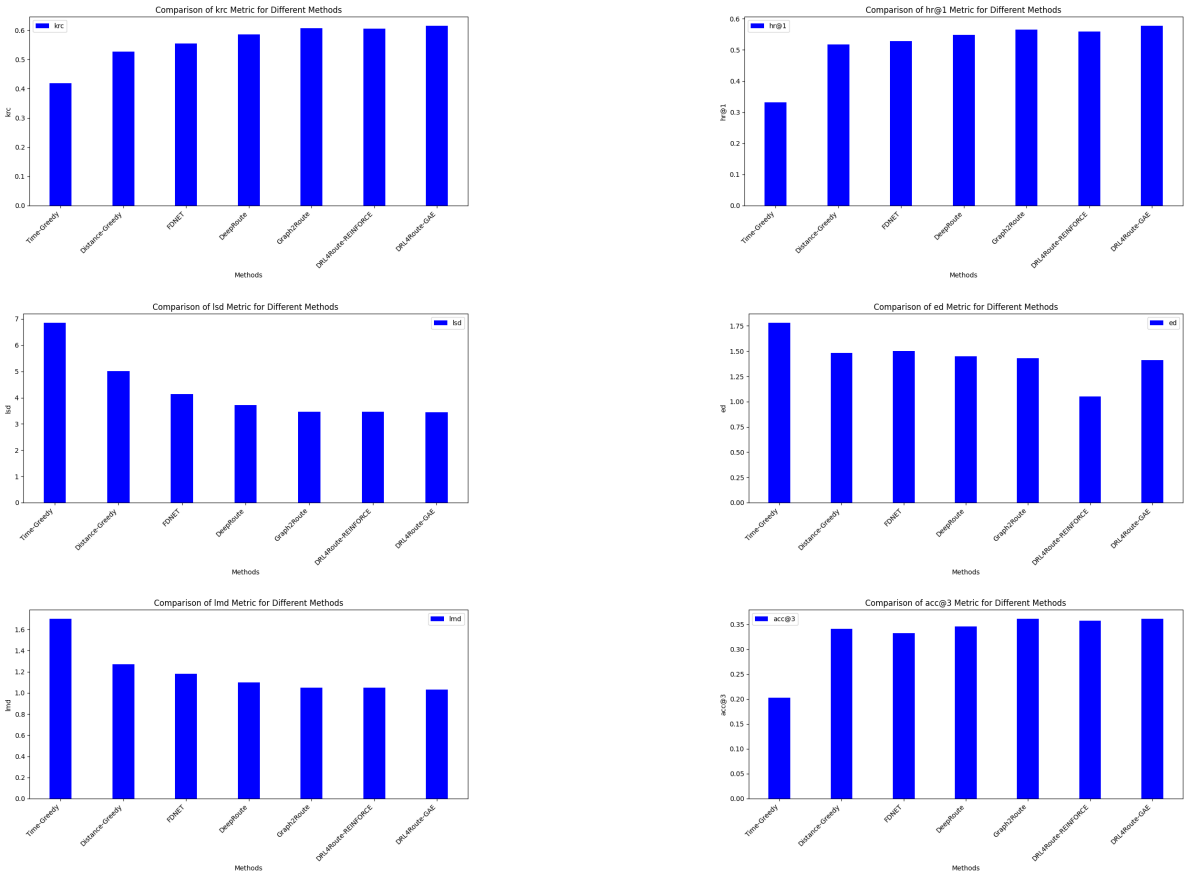

Figure 6.3: Performance of the route prediction methods

7

Managerial Insights

The implementation of Deep Reinforcement Learning models, such as DRL4Route and DRL4Route-GAE, in the tugboat route prediction has showcased the superiority of these models when compared to the traditional heuristic models such as Time and Distance-Greedy models as well as Deep Learning methods such as Deeproute and Graph2Route due to their dynamic learning capabilities and continuously adapting to the new input data. Several managerial insights can be identified by analysing the results from the real-world data provided by KOTUG. These insights not only shed light on the effectiveness of DRL models but also highlight key considerations for future research into this topic.

- **Superior Performance of DRL Models** The DRL4Route and DRL4Route-GAE models outperform traditional heuristic methods such as Time and Distance-Greedy methods as well as other Deep Learning models due to their ability to dynamically learn from the environment and optimise decisions in real time by its Actor-Critic method.
- **Significance of Data Quality and Quantity** The performance of the models is heavily dependent on the quality and quantity of the data. Rich datasets from real-world sources help understand the environment better, leading to a better route prediction overall. The data from KOTUG, while plentiful was not as extensive as the Hangzhou dataset which is the DRL methods perform better than the Deep Learning models when running that data.
- **Effect of Critic's Evaluation Metrics** The choice of the evaluation metric that the critic uses to optimise the route prediction plays a major role in how the model behaves. The model works to maximise these metrics so it stands to reason that if other metrics are chosen as the metrics that the critic optmises, the model would perform differently as it tries to maximise the rewards that the critic assigns to follow those metrics closely. For example, if instead of the KRC metric, the dist metric is used by the Critic, the model's performance based on this metric would improve, leading to a lower value of average distance between the actual and predicted sequences.
- **Scalability and Adaptability Across Environments** The DRL4Route models are not just tailored to a single environment. They are highly scalable and adaptable to various operational contexts. This means that the successful implemen-

tation of this model in one port can be scaled and applied to other ports as well.

- **Room for Improvement in Model Performance** The DRL4Route models currently outperform the other Deep Learning models but this model could be further improved by the incorporation of additional features such as vessel speed, and cargo type. The Critic could be made more complex by adding additional metrics by which the performance of the model is compared. The important thing to note here is not to make the Critic too many things to optimise which results in an overcomplicated Critic which does not perform well.
- **Risk of OVerfitting and Need for Continuous Monitoring** There is a potential risk of overfitting when models are trained on specific datasets and environments. Continuous monitoring and validation of the models across different conditions are crucial to ensure they generalise well and do not become overly reliant on specific patterns in the training data. Regular model audits and performance evaluations are necessary to mitigate this risk.

8

Conclusion

This thesis focused on improving tugboat operations at the Port of Rotterdam using advanced route prediction models. The primary research aim was to enhance the accuracy and efficiency of tugboat scheduling by developing a Deep Reinforcement Learning (DRL) model, DRL4Route, which is capable of optimising the pick-up and drop-off locations for tugboats in a highly dynamic environment. This is crucial for ports like Rotterdam, where the efficient movement of vessels directly impacts operational costs, fuel consumption, and safety.

To achieve this goal, various route prediction techniques mentioned in chapter 5 were evaluated which included traditional methods such as Time-Greedy and Distance-Greedy, as well as more advanced machine learning approaches like FDNet, Graph2Route, and DeepRoute. The Deep Reindorcement Learning models DRL4Route and DRL4Route-GAE were benchmarked against these models using multiple performance metrics, including Kendall Rank Correlation (KRC), Edit Distance (ED), Location Square Deviation (LSD), Location Mean Deviation (LMD), Hit Rate (HR), and Accuracy (ACC). The results indicated that traditional methods like Distance-Greedy and Time-Greedy were outperformed by DRL4Route due to their narrow focus on minimising either time or distance, leading to suboptimal route predictions. Machine learning models, particularly DeepRoute and Graph2Route, demonstrated better performance by utilising spatio-temporal data, but DRL4Route-REINFORCE-GAE emerged as the most accurate. This model was able to directly optimise key metrics using an actor-critic method, leading to superior performance across all of the mentioned testing scenarios. However, when compared using the ”dist” metric against the benchmarks set by KOTUG and the performance of the inverse optimisation model, the inverse optimisation model performed the best, with the lowest average distances between nodes in the predicted route and the actual route. This could be due to the splitting up of the data into the testing, training and validation datasets or how the data was preprocessed. Moreover, changing the critic’s evaluation criterion could alter the way the model performs and possibly improve the performance of the DRL models according to this evaluation metric.

Several key insights were gained through this research. First, the superior performance of DRL models shows the potential of advanced machine learning techniques to replace outdated heuristic methods, which often lead to suboptimal routing. Data

quality emerged as a critical factor, as rich historical data enhanced the model's learning capacity, while poor data could significantly hinder performance. Furthermore, the results underscored the importance of evaluation metrics in determining model success and altering these metrics could lead to different outcomes, depending on operational priorities such as minimising delays or fuel consumption. Scalability and flexibility were also highlighted as major strengths of the DRL models, making them adaptable to other port environments and complex logistical networks. The models' ability to make real-time adjustments offers considerable value in dynamic environments and minimising delays caused by unforeseen changes in vessel schedules or port conditions. Despite these advancements, there remains a need to monitor the risk of overfitting and ensure that models are continuously updated to prevent performance degradation.

Future work

This research lays the foundation for future work in optimising the route prediction of tugboats. Further improvement could be achieved by integrating more features such as vessel speed, cargo type, environmental conditions, and port congestion data. Furthermore, experimenting with different evaluation metrics being used as the metrics that the critic checks the performance of the model against could show different and more accurate results. For example, if the ”dist” evaluation metric discussed in chapter 5 is used as one of the metrics that the critic uses to check the performance of the system, the model could perform in a way that reduces the average distance between the nodes in the predicted route and the actual route. Ultimately, the enhanced safety and resource utilisation achieved through accurate route predictions can lead to substantial economic benefits for both tugboat operators and port authorities.

References

[1] Hervé Abdi. “The Kendall rank correlation coefficient”. In: *Encyclopedia of measurement and statistics* 2 (2007), pp. 508–510.

[2] Andrew G Barto and R Sutton. *Introduction to reinforcement learning*. 1997.

[3] Yi-Ren Chen et al. “RL-routing: An SDN routing algorithm based on deep reinforcement learning”. In: *IEEE Transactions on Network Science and Engineering* 7.4 (2020), pp. 3185–3199.

[4] Jaeyoung Cho et al. “A novel port call optimization framework: A case study of chemical tanker operations”. In: *Applied Mathematical Modelling* 102 (2022), pp. 101–114.

[5] Andrea Conca et al. “Automation in freight port call process: real time data sharing to improve the stowage planning”. In: *Transportation research procedia* 30 (2018), pp. 70–79.

[6] Ruijin Ding et al. “Deep reinforcement learning for router selection in network with heavy traffic”. In: *IEEE Access* 7 (2019), pp. 37109–37120.

[7] Jian Du et al. “Machine Learning-Based Approach to Liner Shipping Schedule Design”. In: *Journal of Shanghai Jiaotong University (Science)* 27.3 (2022), pp. 411–423.

[8] Chengliang Gao et al. “A deep learning method for route and time prediction in food delivery service”. In: *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2021, pp. 2879–2889.

[9] Yuyang Gao et al. “Gnes: Learning to explain graph neural networks”. In: *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2021, pp. 131–140.

[10] Amir Hossein Gharehgozli, Debjit Roy, and René De Koster. “Sea container terminals: New technologies and OR models”. In: *Maritime Economics & Logistics* 18 (2016), pp. 103–140.

[11] Ross Girshick. “Fast r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.

[12] Bruce Golden, Xingyin Wang, and Edward Wasil. “The evolution of the Vehicle Routing Problem—A survey of VRP research and practice from 2005 to 2022”. In: *The Evolution of the Vehicle Routing Problem: A Survey of VRP Research and Practice from 2005 to 2022*. Springer, 2023, pp. 1–64.

[13] MPM Hendriks et al. “Strategic allocation of cyclically calling vessels for multi-terminal container operators”. In: *Flexible Services and Manufacturing Journal* 24 (2012), pp. 248–273.

[14] Jan Hoffmann and S Sirimanne. “Review of maritime transport”. In: *United Nations Conference on Trade and Development*. 2017.

[15] Charles Tapley Hoyt et al. “A unified framework for rank-based evaluation metrics for link prediction in knowledge graphs”. In: *arXiv preprint arXiv:2203.07544* (2022).

[16] YANG Hua-long et al. “Robust Optimization of Vessel Scheduling for Liner Shipping Considering Sea Contingency Time and Collaborative Agreement”. In: *Journal of Transportation Systems Engineering and Information Technology* 21.6 (2021), p. 210.

[17] IMO. *Introduction to imo*. URL: `https://www.imo.org/en/about/pages/default.aspx`.

[18] Liujiang Kang, Qiang Meng, and Kok Choon Tan. “Tugboat scheduling under ship arrival and tugging process time uncertainty”. In: *Transportation Research Part E: Logistics and Transportation Review* 144 (2020), p. 102125.

[19] Grigorios D Konstantakopoulos, Sotiris P Gayialis, and Evripidis P Kechagias. “Vehicle routing problem and related algorithms for logistics distribution: A literature review and classification”. In: *Operational research* 22.3 (2022), pp. 2033–2062.

[20] Kotug. *Harbor Towage*. URL: https://www.kotug.com/towage/harbour-towage/. (accessed: 05.08.2024).

[21] Ki-Beom Lee et al. “Deep reinforcement learning based optimal route and charging station selection”. In: *Energies* 13.23 (2020), p. 6255.

[22] Kai Li, Yongqiang Zhuo, and Xiaoqing Luo. “Optimization method of fuel saving and cost reduction of tugboat main engine based on genetic algorithm”. In: *International Journal of System Assurance Engineering and Management* 13.Suppl 1 (2022), pp. 605–614.

[23] Yiyao Li and William Phillips. “Learning from route plan deviation in last-mile delivery”. In: (2018).

[24] Mikael Lind et al. “The maturity level framework for PortCDM”. In: *Sea Traffic Manage., Norrköping, Sweden, Tech. Rep* 13 (2018).

[25] Xiaowei Mao et al. “Drl4route: A deep reinforcement learning framework for pick-up and delivery route prediction”. In: *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2023, pp. 4628–4637.

[26] Axel Merkel, Joakim Kalantari, and Abdalla Mubder. “Port call optimization and CO2-emissions savings–Estimating feasible potential in tramp shipping”. In: *Maritime Transport Research* 3 (2022), p. 100054.

[27] John Nerbonne, Wilbert Heeringa, and Peter Kleiweg. “Edit distance and dialect proximity”. In: *Time Warps, String Edits and Macromolecules: The theory and practice of sequence comparison* 15 (1999).

[28] Shahrzad Nikghadam et al. “Cooperation between vessel service providers in ports: An impact analysis using simulation for the Port of Rotterdam”. In: *Maritime Transport Research* 4 (2023), p. 100083.

[29] Seong Hyeon Park et al. “Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture”. In: *2018 IEEE intelligent vehicles symposium (IV)*. IEEE. 2018, pp. 1672–1678.

[30] Port-rotterdam. *Facts and Figures*. 2023. URL: https://www.portofrotterdam.com/en/experience-online/facts-and-figures.

[31] René Taudal Poulsen and Helen Sampson. “A swift turnaround? Abating shipping greenhouse gas emissions via port call optimization”. In: *Transportation Research Part D: Transport and Environment* 86 (2020), p. 102460.

[32] Chen Qian and Simon S Lam. “Greedy routing by network distance embedding”. In: *IEEE/ACM Transactions on Networking* 24.4 (2015), pp. 2100–2113.

[33] Filipe Rodrigues and Agostinho Agra. “Berth allocation and quay crane assignment/scheduling problem under uncertainty: A survey”. In: *European Journal of Operational Research* 303.2 (2022), pp. 501–524.

[34] John Schulman et al. “High-dimensional continuous control using generalized advantage estimation”. In: *arXiv preprint arXiv:1506.02438* (2015).

[35] Abhiram Singh, Sidharth Sharma, and Ashwin Gumaste. “Using deep reinforcement learning for routing in IP networks”. In: *2021 International Conference on Computer Communications and Networks (ICCCN)*. IEEE. 2021, pp. 1–9.

[36] Giorgio Stampa et al. “A deep-reinforcement learning approach for software-defined networking routing optimization”. In: *arXiv preprint arXiv:1709.07080* (2017).

[37] Richard S Sutton, Andrew G Barto, et al. *Introduction to reinforcement learning*. Vol. 135. MIT press Cambridge, 1998.

[38] Sander Verduijn. “Identifying the relations between and mapping the processes of the nautical service providers in the Port of Rotterdam”. In: (2017).

[39] Su Wang et al. “Research on the modeling of tugboat assignment problem in container terminal”. In: *Advanced Materials Research* 433 (2012), pp. 1957–1961.

[40] Su Wang et al. “Tugboat scheduling problem based on trust-based ant colony optimization”. In: *Information Computing and Applications: Third International Conference, ICICA 2012, Chengde, China, September 14-16, 2012. Proceedings 3*. Springer. 2012, pp. 373–380.

[41] Xin Wang et al. “An adaptive large neighborhood search algorithm for the tugboat scheduling problem”. In: *Computers & Industrial Engineering* 177 (2023), p. 109039.

[42] Yadong Wang, Qiang Meng, and Peng Jia. “Optimal port call adjustment for liner container shipping routes”. In: *Transportation Research Part B: Methodological* 128 (2019), pp. 107–128.

[43] Xiaoyang Wei et al. “Tugboat scheduling for container ports”. In: *Transportation Research Part E: Logistics and Transportation Review* 142 (2020), p. 102071.

[44] Haomin Wen et al. “Graph2route: A dynamic spatial-temporal graph neural network for pick-up and delivery route prediction”. In: *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*. 2022, pp. 4143–4152.

[45] Haomin Wen et al. “Package pick-up route prediction via modeling couriers’ spatial-temporal behaviors”. In: *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE. 2021, pp. 2141–2146.

[46] Ronald J Williams. “Simple statistical gradient-following algorithms for connectionist reinforcement learning”. In: *Machine learning* 8 (1992), pp. 229–256.

[47] Yiwei Wu et al. “Joint planning of fleet deployment, ship refueling, and speed optimization for dual-fuel ships considering methane slip”. In: *Journal of Marine Science and Engineering* 10.11 (2022), p. 1690.

[48] Yongbin Wu et al. “Research on port tugboat scheduling optimization based on Genetic Algorithm”. In: *2022 2nd International Conference on Computational Modeling, Simulation and Data Analysis (CMSDA)*. IEEE. 2022, pp. 165–170.

[49] Peng Yao, Xingfeng Duan, and Jiale Tang. “An improved gray wolf optimization to solve the multi-objective tugboat scheduling problem”. In: *Plos one* 19.2 (2024), e0296966.

[50] Haocheng Yu. “Tugboat scheduling problem in large container ports: A case study of the Singapore port”. In: *Journal of Physics: Conference Series*. Vol. 1983. 1. IOP Publishing. 2021, p. 012097.

[51] Haifei Zhang et al. “Review of vehicle routing problems: Models, classification and solving algorithms”. In: *Archives of Computational Methods in Engineering* (2022), pp. 1–27.

[52] Yan Zhang et al. “Route prediction for instant delivery”. In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3.3 (2019), pp. 1–25.

[53] Lu Zhen et al. “Integrated berth and yard space allocation under uncertainty”. In: *Transportation Research Part B: Methodological* 162 (2022), pp. 1–27.

[54] Rob Zuidwijk. “Ports and global supply chains”. In: *Ports and Networks*. Routledge, 2017, pp. 26–37.

A

Scientific Paper

Optimising Tugboat Route Prediction Using Deep Reinforcement Learning

Navneet Sajith, Frederik Schulte, Mahnam Saeednia
Faculty of Mechanical Engineering, Delft University of Technology, Delft, The Netherlands

***Abstract*—Tugboats play a crucial role in ensuring the safe and efficient movement of vessels in ports, especially for berthing and unberthing operations. This research focuses on improving tugboat route prediction and scheduling at the Port of Rotterdam by leveraging Deep Reinforcement Learning (DRL). The proposed model, DRL4Route, optimises pick-up and drop-off points by analysing historical data and real-time conditions. It outperforms traditional methods such as Time-Greedy and Distance-Greedy algorithms, which focus narrowly on minimising time or distance. DRL4Route-REINFORCE-GAE emerged as the most accurate method across key performance metrics, including Kendall Rank Correlation (KRC), Edit Distance (ED), and Location Square Deviation (LSD). Future improvements could integrate additional variables such as vessel speed and environmental conditions to further refine the model's predictions. The research underscores the potential of DRL in optimising complex maritime logistics, ultimately contributing to a more resilient and sustainable port ecosystem.**

***Index Terms*—Tugboat Scheduling, Route Prediction, Deep Reinforcement Learning, Port Operations, Maritime Logistics, DRL4Route**

I. Introduction

Ports are indispensable to global trade, facilitating the transport of approximately 80% of global goods [44]. Their role in the transshipment of cargo between ships and land-based transport systems is crucial for the global supply chain's efficiency [45]. The global maritime industry, while resilient, has faced unprecedented disruptions from events such as the COVID-19 pandemic and ongoing geopolitical conflicts, including the wars in Ukraine and Palestine. Nevertheless, the industry is projected to grow by 2.4% in 2023 and 2% annually from 2024 to 2028, highlighting the ongoing importance of maritime trade despite global challenges [46].

The Port of Rotterdam, Europe's largest port, serves as a vital hub within this ecosystem. It handles over 438 billion tons of cargo annually, contributing €38.6 billion to the Dutch economy—about 3.2% of the Netherlands' GDP [1]. Efficient port call optimisation is essential to manage this immense flow of goods, as illustrated by the breakdown of cargo throughput in Fig. 1. The ability to synchronise vessel movements with resource allocation, while minimising delays caused by dynamic vessel schedules, unpredictable weather conditions, and infrastructure constraints, is critical to maintaining efficient operations.

The coordination of multiple actors—ranging from port authorities, vessel operators, and pilots, to tugboat companies, mooring crews, and terminal operators—plays a pivotal role

(Gross weight x 1,000 tonnes)	2023	2022	Difference (number)	Difference (%)
Dry bulk cargo	70,642	80,064	-9,422	-11.8%
Liquid bulk cargo	205,627	212,771	-7,144	-3.4%
Total bulk cargo	**276,269**	**292,835**	**-16,566**	**-5.7%**
Containers	130,162	139,657	-9,495	-6.8%
Break bulk	32,371	34,889	-2,518	-7.2%
Total general cargo	**162,533**	**174,546**	**-12,013**	**-6.9%**
Total cargo throughput	**438,802**	**467,381**	**-28,579**	**-6.1%**
Total numbers of containers	7,816,755	8,315,417	-498,662	-6.0%
Total TEUs	13,446,709	14,456,313	-1,009,604	-7.0%

Fig. 1. Throughput in Port of Rotterdam [1]

in ensuring that the port functions smoothly. The intricate interaction of these stakeholders is visualised in Fig.2, which demonstrates the complex sequence of events required for a vessel's safe arrival, berthing, and departure. This "Port Call Metro Map" illustrates the necessity for real-time data sharing and precise timing across all involved parties to avoid delays, reduce vessel waiting times, and optimise the overall port call process.

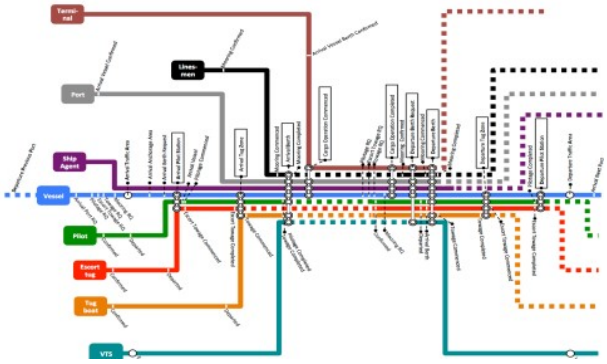

Fig. 2. Port Call Metro Map [47]

Tugboats are a critical element of this system, assisting large vessels in safely navigating the narrow and often shallow waters of ports [48]. Pilots, who oversee a ship's navigation, determine the quantity and duration of tugboat assistance needed [57]. The Port of Rotterdam handles approximately 30,000 sea-going vessels annually, with tugboats ensuring the safe and timely berthing and unberthing of these ships. Tugboat operations are primarily handled by private companies, such as Kotug International BV, whose services include tugboat design, building, chartering, and operations. Kotug assigns tugboats to assist sea-going vessels entering or leaving the port, with a focus on ensuring safety and efficiency in each maneuver. However, one of the key challenges faced by these

companies is the optimisation of tugboat operations within the port's intricate and busy environment.

In the Port of Rotterdam, the scheduling and coordination of tugboat operations is not standardised, leading to inefficiencies. Tugboats are sometimes connected to vessels at different locations from their planned assignments, adding complexity to the already challenging task of optimising vessel movements within the port. This challenge is akin to solving a pick-up and delivery service problem, where deviations from the planned routes can significantly impact efficiency. Studies have shown that actual routes taken by pilots often differ from those predicted by routing tools, with similar deviations observed in related logistics fields. For example, in a study conducted on delivery operations for a large soft drinks company in the US and Mexico, three out of four deliveries deviated from the planned route [36], [38]. Such discrepancies underscore the importance of accurate route prediction for enhancing port operations.

This research explores the potential of using advanced machine learning techniques, specifically a deep reinforcement learning (DRL) framework called DRL4Route, to optimise tugboat pick-up locations and route planning. DRL combines Deep Learning (DL) and Reinforcement Learning (RL) to create models that can make decisions in complex, dynamic environments [17]. DL is used to identify patterns in data, while RL trains agents to perform tasks by rewarding actions that contribute to improved performance. The goal is to maximise the overall system reward, continuously learning and adapting to the environment to improve decision-making [35].

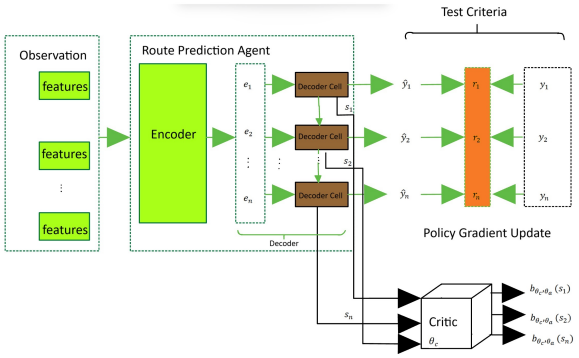

Fig. 3. Framework of DRL4Route

As illustrated in Fig. 3, DRL4Route employs an Actor-Critic architecture, where the model observes features of the environment, processes them through an encoder, and generates route predictions through a series of decoder cells. The model then compares the predicted route with the actual route taken, using this comparison to compute rewards. These rewards are used to update the route prediction agent's policies, enabling it to improve its predictions over time. The continuous learning capabilities of DRL make it well-suited to dynamic environments like the Port of Rotterdam, where variables such as vessel schedules, weather conditions, and infrastructure limitations are constantly changing.

The application of DRL to route prediction has shown promising results in other fields. For instance, Trailnet, a DRL-based model for routing in IP networks, replaced traditional forwarding tables with a computational model trained to minimise packet forwarding costs across different ports [40]. Similarly, DRL has been used in software-defined networks (SDNs) to optimise traffic routing, significantly improving network performance compared to traditional methods [39]. These case studies highlight the potential of DRL to enhance complex routing and scheduling problems, making it a promising approach for optimising tugboat operations in the Port of Rotterdam.

This research aims to develop a novel route prediction method for Kotug's tugboat operations, leveraging DRL to optimise both pick-up locations and route planning. By improving route prediction accuracy and reducing the discrepancies between planned and actual routes, this research is expected to enhance Kotug's operational efficiency, lower fuel consumption, reduce emissions, and minimise costs. Ultimately, this work seeks to provide Kotug with the necessary tools to navigate the complex and dynamic environment of the Port of Rotterdam, while contributing to a more efficient and sustainable port ecosystem.

II. Literature Survey

Port call optimisation is essential to enhancing maritime logistics, ensuring that processes like berth allocation, tugboat scheduling, and route prediction are efficient as global trade volumes rise. Ports such as Rotterdam, with their growing number of ships, benefit significantly from optimisation efforts aimed at improving resource allocation and reducing delays. This section reviews recent advancements in port call optimisation, particularly in the context of tugboat operations, highlighting relevant studies.

Category	Paper References
Berth Allocation, Scheduling, and Port Infrastructure	Rodrigues and Agra [3], Conca et al. [4], Hendriks et al. [49], Gharehgozli et al.[50], Zhen et al.[51]
Route Prediction and Optimisation	Wang et al. [7], Cho et al. [8], Yang et al. [9], Du et al. [10], Mao et al. [12], Qian [19], Wen [18], Graph2Route Wen [16], OSquare Zhang [15]
Fleet Management and Operational Efficiency	Wu et al. [11], Li et al. [2], Merkel et al. [5], Poulsen and Sampson [6]
Tugboat Scheduling	Wang et al. [13], Wei et al. [14], Yao et al.[54], Yu.[55]

TABLE I
Categorisation of Papers in Port Call Optimisation

Table I organises relevant studies into categories addressing different aspects of port call optimisation, helping illustrate how various components contribute to the overall process.

A. Berth Allocation, Scheduling, and Port Infrastructure

Berth allocation and scheduling are critical to minimising delays in port operations. Rodrigues and Agra [3] explore berth allocation and quay crane scheduling under uncertainty, proposing strategies to mitigate disruptions caused by variable

ship arrival times. Conca et al. [4] emphasise the importance of real-time data sharing for better coordination among stakeholders. Additionally, investments in port infrastructure, as discussed by Hendriks et al. [49], are necessary to enhance long-term port efficiency. Studies by Gharehgozli et al. [50] and Zhen et al. [51] focus on the integration of sea and land-side operations to reduce congestion and streamline processes.

B. Route Prediction and Optimisation

Accurate route prediction and optimisation are crucial for improving the allocation of resources such as tugboats in ports. Wang et al. [7] and Cho et al. [8] provide optimisation models for shipping routes that can be adapted for tugboat operations. Mao et al. [12] introduce DRL4Route, a deep reinforcement learning framework that adapts to real-time conditions, making it suitable for dynamic environments like the Port of Rotterdam. Graph2Route, proposed by Wen [16], leverages spatio-temporal graphs to enhance real-time route prediction, offering potential for more efficient tugboat coordination.

C. Fleet Management and Operational Efficiency

Effective fleet management is vital for reducing delays and maximising fuel efficiency. Wu et al. [11] address fleet deployment strategies, while Li et al. [2] focus on optimising fuel consumption for tugboats, contributing to both economic and environmental sustainability. Merkel et al. [5] and Poulsen and Sampson [6] emphasise the importance of optimising port calls to reduce emissions, aligning operational efficiency with environmental goals.

D. Tugboat Scheduling

Tugboat scheduling is essential for guiding large vessels through congested waterways. Yao et al. [54] propose the Improved Grey Wolf Optimisation algorithm to address multi-objective scheduling problems, while Yu [55] presents a mixed-integer linear programming model for minimising processing costs. These models, when adapted to ports like Rotterdam, can enhance the efficiency of tugboat operations and reduce delays.

The reviewed studies highlight the importance of integrating machine learning and optimisation algorithms to improve tugboat operations. By leveraging these advancements, ports like Rotterdam can significantly enhance efficiency, reduce costs, and improve environmental sustainability.

III. Problem Definition

The towage process in the Port of Rotterdam is essential for the safe and efficient movement of vessels, particularly during berthing and unberthing. Tugboats assist large vessels in navigating the port's confined waters, ensuring precise control to prevent accidents or collisions. However, current methods for assigning tugboat routes rely heavily on human expertise and static vehicle routing programs, which lack the flexibility to adapt to real-time changes such as vessel delays or environmental factors [32] [31]. As port complexity increases, these systems often struggle, leading to delays and higher costs [33].

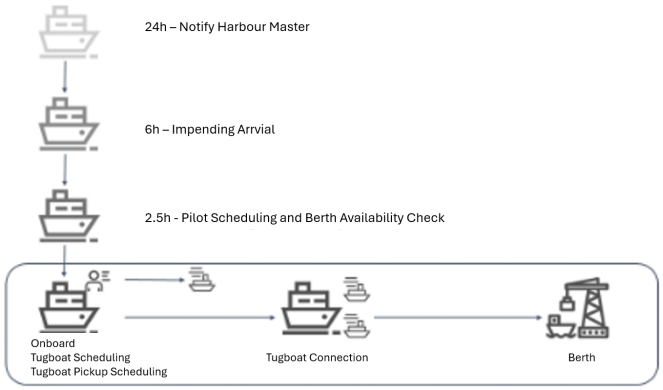

Fig. 4. Tugboat pick-up process

The towage process begins well before a vessel's arrival, with scheduling steps involving the Harbour Master, pilots, and tugboats. Despite extensive planning, predicting the exact connection and disconnection points for tugboats remains a challenge. KOTUG, which manages these operations using the Kotug Optiport scheduling tool, often faces discrepancies between predicted and actual towage locations. This misalignment, driven by pilots making real-time decisions, leads to inefficiencies such as delays, increased fuel consumption, and suboptimal resource use.

The core issue is the difficulty in predicting tugboat connection/disconnection points due to the dynamic port environment and varying vessel capabilities. These inaccuracies result in operational delays, higher costs, and reduced port efficiency.

This research aims to develop a more accurate predictive model, DRL4Route, leveraging Deep Reinforcement Learning (DRL) to improve real-time flexibility and the precision of pick-up/drop-off predictions. By incorporating historical towage data and real-time adjustments, the model will enhance towage operations, reduce delays, and lower operational costs, contributing to improved economic performance for KOTUG and the Port of Rotterdam.

The objective of this study is to address the challenges in predicting towage locations and optimise tugboat schedules through advanced machine learning techniques, ultimately improving safety, efficiency, and profitability for all stakeholders.

IV. Methodology

A. Data and Preprocessing

The dataset used for this research was provided by KOTUG and covers tugboat operations between May 31, 2022, and May 31, 2023. The dataset contains 42,232 entries across 15 columns, including numerical and categorical data relevant to tugboat movements.

The key preprocessing steps included:

- **Handling Missing Values:** The columns "From Berth," "To Berth," "From Haven," and "To Haven" contained a significant number of missing entries. Missing values were filled with a placeholder "Unknown" to retain the records for further analysis.

- **Datetime Conversion:** Date columns were converted to a standard "datetime" format to enable time-based operations, such as calculating the duration of tugboat operations and analysing time trends.
- **Geolocation Standardisation:** Columns containing geographical coordinates ("From Location X," "From Location Y," "To Location X," and "To Location Y") were checked for consistency. All coordinates were correctly formatted as floating-point numbers, eliminating the need for further standardisation.

The processed dataset was then used for the route prediction problem, ensuring that no missing data remained and that all relevant columns were formatted consistently for use in machine learning models.

Data preprocessing involved handling missing values, standardising geographical data, and converting timestamps to a consistent format. Missing values in location data were filled with placeholders to retain as much information as possible. The resulting dataset was then used as input for the predictive model.

B. Model Architecture

The DRL4Route framework employs an encoder-decoder architecture for route prediction. The encoder processes historical tugboat movement data and converts it into a fixed-sise context vector, which summarises the current state of the system. The decoder then generates future route predictions based on the context vector and real-time input data. Both the encoder and decoder are built using Long Short-Term Memory (LSTM) layers to handle the sequential nature of tugboat operations.

DRL4Route DRL4Route is a Deep Reinforcement Learning (DRL) model developed to predict tugboat routes by optimising pick-up and drop-off points. It treats route prediction as a sequential decision-making process, where the goal is to maximise cumulative rewards based on the efficiency of the route.

1) Reinforcement-Guided Route Prediction: The route prediction problem is framed as a Markov Decision Process (MDP), where:

- S represents the states (i.e., the current system status).
- A represents the actions (i.e., the route decisions for the tugboat).
- R is the reward function, evaluating the performance of each action.

The goal is to maximise the cumulative reward, with the agent learning the optimal policy to minimise delays and improve route efficiency. The cumulative reward function is given by:

$$\theta^* = \arg\max_{\theta} E_{\pi_\theta}\left[\sum_{t=1}^{T} \gamma^t r_t\right] \tag{1}$$

2) Actor-Critic Architecture: To enhance stability and reduce variance in policy learning, DRL4Route employs an Actor-Critic architecture. The Actor selects actions based on the current state, while the Critic evaluates the actions by estimating the state-value function $V(s_t)$ and the state-action value $Q(s_t, a_t)$. The advantage function $A(s_t, a_t)$ measures how much better an action is compared to the baseline, helping the Actor improve its decisions:

$$A^{\pi_{\theta_a}}(s_t, a_t) = r_t + \gamma V_{\pi_{\theta_a}}(s_{t+1}) - V_{\pi_{\theta_a}}(s_t) \tag{2}$$

The loss function for training the Actor is:

$$L_{\theta_a} = \frac{1}{N}\sum_{i=1}^{N}\sum_{t\in T} \log \pi_{\theta_a}(a_{i,t} \mid s_{i,t}) A_{\pi_{\theta_a}}(s_{i,t}, a_{i,t}) \tag{3}$$

3) Training and Loss Functions: The Critic evaluates the state-value function, while the Actor updates the policy using policy gradient methods. The Critic is trained with a regression loss that minimises the error between the predicted state value and the actual reward:

$$L_{\theta_c} = \frac{1}{N}\sum_{i=1}^{N}\sum_{t\in T} \text{smooth}L_1(b_{\theta_c,\theta_a}(s_{i,t}) - r(\hat{y}_{i,t}, \cdots, \hat{y}_{i,n})) \tag{4}$$

Initially, the system is pre-trained using cross-entropy loss to maximise the likelihood of correct predictions. This is followed by joint training of both the Actor and Critic:

$$L_{\theta_{ac}} = \alpha_{\theta_c} L_{\theta_c} + \alpha_{\theta_a} L_{\theta_a} + \alpha_{CE} L_{CE} \tag{5}$$

4) Generalised Advantage Estimation: Generalised Advantage Estimation (GAE) is used to balance the trade-off between bias and variance in policy estimation. It allows the system to compute a more stable estimate of the advantage function by adjusting for future rewards:

$$A^{\text{GAE}(\gamma,\lambda)}_{\pi_{\theta_a}}(s_t, a_t) = \sum_{t'=t}^{T} (\gamma\lambda)^{(t'-t)} \Big(r(s_{t'}, a_{t'}) + \gamma V_{\pi_{\theta_a}}(s_{t'+1}) - V_{\pi_{\theta_a}}(s_{t'}) \Big) \tag{6}$$

5) Encoder-Decoder in DRL4Route: The Encoder-Decoder architecture is used to handle the sequential nature of route prediction. The Encoder processes spatio-temporal features of unfinished tasks, generating a fixed-size context vector E, while the Decoder predicts the next route step based on this context and previous outputs.

The attention mechanism ensures the Decoder focuses on relevant tasks, dynamically adjusting the predictions step-by-step until the entire route is generated:

$$o_t, h_{t+1} = \text{DecoderCell}(E, C, h_t, \hat{Y}_{1:t-1}) \tag{7}$$

V. Evaluation Metrics

The performance of the DRL4Route framework is evaluated using several metrics:

A. Kendall Rank Correlation (KRC)

KRC measures the ordinal association between predicted and actual routes. A higher KRC indicates a greater similarity between the predicted route ranking and the actual route ranking.

B. Edit Distance (ED)

Edit distance is used to calculate the minimum number of operations required to transform the predicted route into the actual route. Lower ED values signify more accurate predictions.

C. Location Mean Deviation (LMD) and Location Square Deviation (LSD)

These metrics quantify the deviation between the predicted and actual pick-up/drop-off locations. The smaller the deviation, the more accurate the model.

D. Hit Rate @k and Accuracy @k

These metrics evaluate how many of the top-k predicted routes match the actual routes. Hit Rate @k measures the proportion of correctly predicted routes, while Accuracy @k considers the order of the predictions.

E. Dist

Dist measures the average physical distance between corresponding points on the predicted and actual routes. This is particularly relevant for optimising tugboat fuel consumption and scheduling.

VI. Results

Tables for eval_max = 11 and 25

eval_max = 11

krc	ed	lsd	lmd	hr@1	acc@3	method
0.548	1.502	2.775	1.032	0.523	0.290	DR
0.416	1.580	4.176	1.332	0.417	0.230	DG
0.385	1.586	3.531	1.232	0.474	0.214	FN
0.530	1.519	2.834	1.056	0.509	0.276	G2R
0.386	1.804	4.417	1.528	0.304	0.127	OS
0.445	1.715	3.914	1.330	0.369	0.198	TG
0.549	1.487	2.595	1.002	0.521	0.294	DRL-R
0.550	1.494	2.573	1.003	0.522	0.296	DRL-R-GAE

eval_max = 25

krc	ed	lsd	lmd	hr@1	acc@3	method
0.548	1.519	2.801	1.036	0.523	0.290	DR
0.415	1.598	4.216	1.337	0.416	0.230	DG
0.385	1.586	3.531	1.232	0.474	0.214	FN
0.530	1.536	2.860	1.060	0.509	0.276	G2R
0.385	1.830	4.485	1.535	0.303	0.127	OS
0.444	1.734	3.947	1.335	0.368	0.197	TG
0.548	1.504	2.623	1.007	0.521	0.294	DRL-R
0.550	1.511	2.601	1.006	0.521	0.296	DRL-R-GAE

TABLE II
COMPARISON OF DIFFERENT METHODS FOR EVAL_MAX 11 AND 25

Table II presents the performance of different route prediction methods. Time and Distance-Greedy methods underperform, as they focus only on reducing immediate time or distance, failing to optimise the overall goal. Osquare, despite using XGBoost, also lags behind the deep learning models. Among the deep learning models, Graph2Route outperforms FDNet, as its graph-based architecture better captures spatio-temporal dependencies. Deeproute performs slightly better than Graph2Route, benefiting from its powerful transformer encoder and attention mechanism.

DRL4Route-REINFORCE surpasses other models by directly optimising the solution based on evaluation metrics. This addresses the issues faced by cross-entropy-based methods. The best-performing model, DRL4Route-REINFORCE-GAE, balances bias and variance using the Generalised Advantage Estimator (GAE), resulting in the most accurate predictions.

Method	Incoming harbour EURO	Outgoing harbour EURO	Incoming berth EUROPAH
DRL4Route	1.2091	1.7904	1.9717
DRL4RouteGAE	1.2090	1.79	1.9716
Inverse Optimisation	1.1780	1.7860	1.4500
Benchmark (Kotug)	1.1020	2.1760	1.5170

TABLE III
COMPARISON OF AVERAGE DISTANCE BETWEEN POINTS IN PREDICTED VS ACTUAL ROUTE - DIST

III compares the average distance between predicted and actual routes for various models. DRL4Route and DRL4RouteGAE outperform the KOTUG benchmark but do not outperform the inverse optimisation model in some cases. This suggests that while DRL models are effective, there is room for improvement in specific cases.

Results in IV show DRL4Route and DRL4RouteGAE models consistently outperform traditional methods in real-world applications. These models have higher KRC, Hit Rate (HR@1), and Accuracy (ACC@3) scores, indicating that they closely match predicted routes with actual outcomes. These models are thus highly suitable for route prediction tasks in dynamic environments.

krc	ed	lsd	lmd	hr@1	acc@3	method
0.4192	1.78	6.85	1.70	0.3315	0.2032	Time-Greedy
0.5268	1.48	5.02	1.27	0.5168	0.3413	Distance-Greedy
0.5547	1.50	4.14	1.18	0.5276	0.3322	FDNET
0.5861	1.45	3.71	1.10	0.5476	0.3464	DeepRoute
0.6063	1.43	3.47	1.05	0.5645	0.3612	Graph2Route
0.6057	1.05	3.47	1.05	0.5585	0.3574	DRL4Route-REINFORCE
0.6147	1.41	3.44	1.03	0.5772	0.3612	DRL4Route-GAE

TABLE IV
PERFORMANCE COMPARISON OF VARIOUS METHODS ON DIFFERENT METRICS

As shown in Table IV and II, the DRL4Route-REINFORCE-GAE model achieved the highest Kendall Rank Correlation (KRC), the lowest Edit Distance (ED), and superior Hit Rate @1, demonstrating its ability to predict tugboat routes more accurately than other models. It also outperformed the FDNet and Graph2Route models in terms of Location Mean Deviation (LMD) and Location Square Deviation (LSD), indicating a closer match between the predicted and actual tugboat routes.

VII. Discussion

The performance of the DRL4Route model shows significant improvement over traditional and deep learning-based

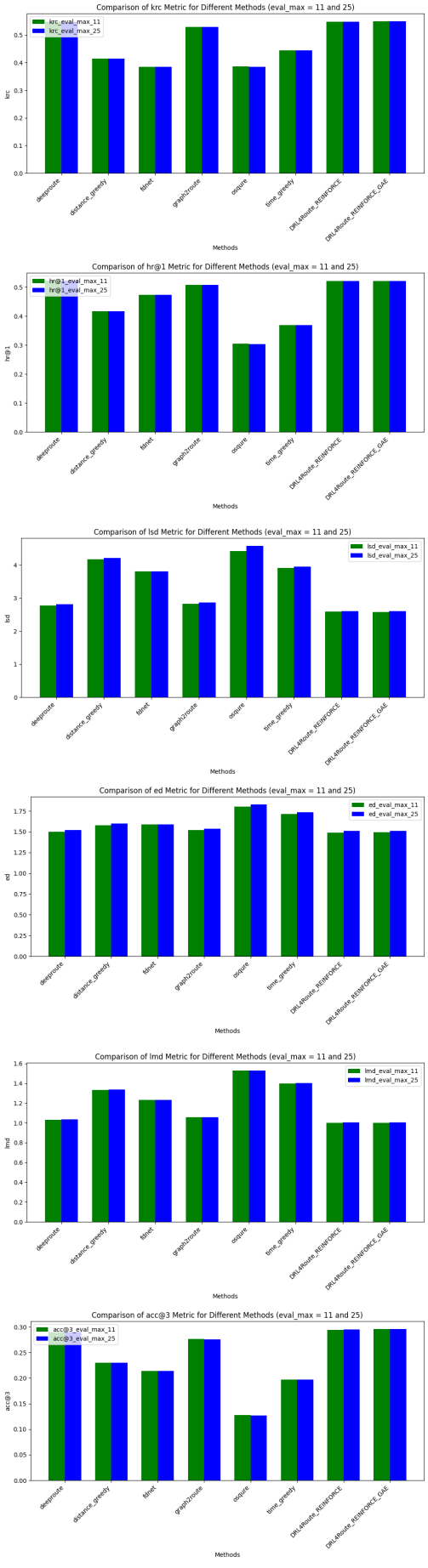

Fig. 5. Performance of route prediction methods

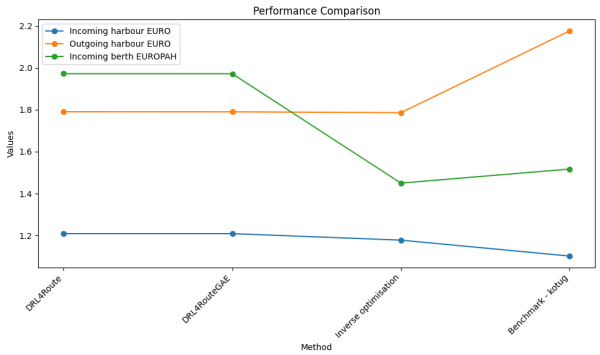

Fig. 6. Average distance between predicted and actual nodes

methods for route prediction in dynamic environments such as port operations. The reinforcement learning component enables the model to continuously learn from real-time feedback, adapting to changing conditions more effectively than static or predefined models.

Several key factors contributed to the superior performance of DRL4Route:

- **Real-Time Adaptability**: The model's ability to adapt to real-time changes in port conditions allowed for more accurate route predictions even under unpredictable circumstances.
- **Optimised Resource Allocation**: By predicting more accurate tugboat routes, the model helped optimise the allocation of resources, leading to reduced fuel consumption and operational costs.
- **Generalied Advantage Estimation (GAE)**: The use of GAE reduced the bias-variance tradeoff, allowing the model to generalie better to unseen data while maintaining high accuracy.

A. *Limitations*

While DRL4Route showed promising results, several limitations must be considered:

- **Data Dependency**: The model's performance is heavily reliant on the quality and quantity of historical tugboat operation data. In cases where the data is sparse or noisy, the model may produce suboptimal predictions.
- **Computational Complexity**: Training a deep reinforcement learning model can be computationally expensive and time-consuming. Real-time deployment of such models in high-traffic ports may require substantial computational resources.
- **Scalability**: Although DRL4Route performed well in the context of the Port of Rotterdam, the model may require further fine-tuning and retraining to generalie to other port environments with different operational constraints.

VIII. Conclusion

This thesis explored improving tugboat operations at the Port of Rotterdam through advanced route prediction models. The primary objective was to enhance the accuracy and efficiency

of tugboat scheduling using a Deep Reinforcement Learning (DRL) model, DRL4Route, aimed at optimising pick-up and drop-off locations in a dynamic environment. Efficient tugboat movement is crucial in ports like Rotterdam, as it directly impacts operational costs, fuel consumption, and safety.

Various prediction techniques, as outlined in "V", were assessed including traditional methods like Time-Greedy and Distance-Greedy, and machine learning approaches such as FDNet, Graph2Route, and DeepRoute. DRL models, particularly DRL4Route and DRL4Route-GAE, were evaluated against these methods using performance metrics like Kendall Rank Correlation (KRC), Edit Distance (ED), Location Square Deviation (LSD), Location Mean Deviation (LMD), Hit Rate (HR), and Accuracy (ACC). Traditional methods, such as Distance-Greedy, were outperformed by DRL4Route due to their limited focus on either time or distance, leading to suboptimal routing. Machine learning methods like DeepRoute and Graph2Route demonstrated better performance, but DRL4Route-REINFORCE-GAE proved to be the most accurate overall, as it optimised key metrics through an actor-critic method. However, when compared to the inverse optimisation model and KOTUG benchmarks using the "dist" metric, the inverse optimisation model showed the lowest average distances between predicted and actual routes, suggesting that further improvements could be achieved by refining the data preprocessing and evaluation processes.

Key insights from the research highlight the superior performance of DRL models over traditional heuristics and machine learning methods. DRL models' capacity to adapt and optimise based on real-time conditions showed clear benefits in dynamic port environments. Data quality and the selection of evaluation metrics were crucial in model success, with the possibility that altering these metrics could yield different optimisation results depending on operational priorities. Although the results were promising, monitoring overfitting and ensuring model updates remain essential to maintaining long-term model performance.

Future Work

Future research can expand on these results by incorporating additional features like vessel speed, cargo type, environmental conditions, and port congestion. Adjusting evaluation metrics for the critic's performance, such as integrating the "dist" metric from V, could refine the model's route accuracy. With continued development, DRL models can offer substantial economic and operational benefits to tugboat operators and port authorities by improving safety, reducing delays, and optimising resource utilisation.

REFERENCES

[1] Port-rotterdam, "Facts and Figures," 2023. Available: https://www.portofrotterdam.com/en/experience-online/facts-and-figures. [Accessed: Mar. 20, 2024].

[2] K. Li, Y. Zhuo, and X. Luo, "optimisation method of fuel saving and cost reduction of tugboat main engine based on genetic algorithm," *International Journal of System Assurance Engineering and Management*, vol. 13, suppl. 1, pp. 605–614, 2022.

[3] F. Rodrigues and A. Agra, "Berth allocation and quay crane assignment/scheduling problem under uncertainty: A survey," *European Journal of Operational Research*, vol. 303, no. 2, pp. 501–524, 2022.

[4] A. Conca, A. Di Febbraro, D. Giglio, and F. Rebora, "Automation in freight port call process: real-time data sharing to improve the stowage planning," *Transportation Research Procedia*, vol. 30, pp. 70–79, 2018.

[5] A. Merkel, J. Kalantari, and A. Mubder, "Port call optimisation and CO2-emissions savings–Estimating feasible potential in tramp shipping," *Maritime Transport Research*, vol. 3, p. 100054, 2022.

[6] R. Poulsen and H. Sampson, "A swift turnaround? Abating shipping greenhouse gas emissions via port call optimisation," *Transportation Research Part D: Transport and Environment*, vol. 86, p. 102460, 2020.

[7] Y. Wang, Q. Meng, and P. Jia, "Optimal port call adjustment for liner container shipping routes," *Transportation Research Part B: Methodological*, vol. 128, pp. 107–128, 2019.

[8] J. Cho, B. Craig, M. Hur, and G. J. Lim, "A novel port call optimisation framework: A case study of chemical tanker operations," *Applied Mathematical Modelling*, vol. 102, pp. 101–114, 2022.

[9] Y. Hua-long, S. Zhao, F. Xu, and J. Duan, "Robust optimisation of Vessel Scheduling for Liner Shipping Considering Sea Contingency Time and Collaborative Agreement," *Journal of Transportation Systems Engineering and Information Technology*, vol. 21, no. 6, p. 210, 2021.

[10] J. Du, X. Zhao, L. Guo, and J. Wang, "Machine Learning-Based Approach to Liner Shipping Schedule Design," *Journal of Shanghai Jiaotong University (Science)*, vol. 27, no. 3, pp. 411–423, 2022.

[11] Y. Wu, Y. Huang, H. Wang, and L. Zhen, "Joint planning of fleet deployment, ship refueling, and speed optimisation for dual-fuel ships considering methane slip," *Journal of Marine Science and Engineering*, vol. 10, no. 11, p. 1690, 2022.

[12] X. Mao, H. Wen, H. Zhang, H. Wan, L. Wu, J. Zheng, H. Hu, and Y. Lin, "DRL4Route: A deep reinforcement learning framework for pick-up and delivery route prediction," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4628–4637, 2023.

[13] X. Wang, Y. Liang, X. Wei, and E. P. Chew, "An adaptive large neighborhood search algorithm for the tugboat scheduling problem," *Computers & Industrial Engineering*, vol. 177, p. 109039, 2023.

[14] X. Wei, S. Jia, Q. Meng, and K. C. Tan, "Tugboat scheduling for container ports," *Transportation Research Part E: Logistics and Transportation Review*, vol. 142, p. 102071, 2020.

[15] Y. Zhang, Y. Liu, G. Li, Y. Ding, N. Chen, H. Zhang, T. He, and D. Zhang, "Route prediction for instant delivery," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, no. 3, pp. 1–25, 2019.

[16] H. Wen, Y. Lin, X. Mao, F. Wu, Y. Zhao, H. Wang, J. Zheng, L. Wu, H. Hu, and H. Wan, "Graph2Route: A dynamic spatial-temporal graph neural network for pick-up and delivery route prediction," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4143–4152, 2022.

[17] C. Gao, F. Zhang, G. Wu, Q. Hu, Q. Ru, J. Hao, R. He, and Z. Sun, "A deep learning method for route and time prediction in food delivery service," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 2879–2889, 2021.

[18] H. Wen, Y. Lin, F. Wu, H. Wan, S. Guo, L. Wu, C. Song, and Y. Xu, "Package pick-up route prediction via modeling couriers' spatial-temporal behaviors," in *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pp. 2141–2146, 2021.

[19] C. Qian and S. S. Lam, "Greedy routing by network distance embedding," *IEEE/ACM Transactions on Networking*, vol. 24, no. 4, pp. 2100–2113, 2015.

[20] H. Abdi, "The Kendall rank correlation coefficient," *Encyclopedia of Measurement and Statistics*, vol. 2, pp. 508–510, 2007.

[21] J. Nerbonne, W. Heeringa, and P. Kleiweg, "Edit distance and dialect proximity," in *Time Warps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison*, vol. 15, CSLI Press, 1999.

[22] C. T. Hoyt, M. Berrendorf, M. Galkin, V. Tresp, and B. M. Gyori, "A unified framework for rank-based evaluation metrics for link prediction in knowledge graphs," *arXiv preprint arXiv:2203.07544*, 2022.

[23] A. G. Barto and R. S. Sutton, *Introduction to Reinforcement Learning*, MIT Press, 1997.

[24] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, pp. 229–256, 1992.

[25] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*, vol. 135, MIT Press, 1998.

[26] R. Girshick, "Fast R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1440–1448, 2015.

[27] J. Li, X. Duan, Z. Xiong, and P. Yao, "Tugboat Scheduling Method Based on the NRPER-DDPG Algorithm: An Integrated DDPG Algorithm with Prioritized Experience Replay and Noise Reduction," *Sustainability*, vol. 16, no. 8, p. 3379, 2024.

[28] J. Li, X. Duan, Z. Xiong, and P. Yao, "Tugboat Scheduling Method Based on the NRPER-DDPG Algorithm: An Integrated DDPG Algorithm with Prioritized Experience Replay and Noise Reduction," *Sustainability*, vol. 16, no. 8, p. 3379, 2024.

[29] S. M. Lee, J. H. Lee, M. I. Roh, K. S. Kim, S. H. Ham, and H. W. Lee, "An optimisation model of tugboat operation for conveying a large surface vessel," *Journal of Computational Design and Engineering*, vol. 8, no. 2, pp. 654–675, 2021.

[30] C. Sun, M. Li, L. Chen, and P. Chen, "Dynamic Tugboat Scheduling for Large Seaports with Multiple Terminals," *Journal of Marine Science and Engineering*, vol. 12, no. 1, p. 170, 2024.

[31] G. D. Konstantakopoulos, S. P. Gayialis, and E. P. Kechagias, "Vehicle routing problem and related algorithms for logistics distribution: A literature review and classification," *Operational Research*, vol. 22, no. 3, pp. 2033–2062, 2022.

[32] B. Golden, X. Wang, and E. Wasil, "The evolution of the Vehicle Routing Problem—A survey of VRP research and practice from 2005 to 2022," in *The Evolution of the Vehicle Routing Problem: A Survey of VRP Research and Practice from 2005 to 2022*, pp. 1–64, Springer, 2023.

[33] H. Zhang, H. Ge, J. Yang, and Y. Tong, "Review of vehicle routing problems: Models, classification and solving algorithms," *Archives of Computational Methods in Engineering*, pp. 1–27, 2022.

[34] Y. Gao, T. Sun, R. Bhatt, D. Yu, S. Hong, and L. Zhao, "GNES: Learning to explain graph neural networks," in *2021 IEEE International Conference on Data Mining (ICDM)*, pp. 131–140, 2021.

[35] K. B. Lee, M. A. Ahmed, D. K. Kang, and Y. C. Kim, "Deep reinforcement learning based optimal route and charging station selection," *Energies*, vol. 13, no. 23, p. 6255, 2020.

[36] Y. Li and W. Phillips, "Learning from route plan deviation in last-mile delivery," unpublished, 2018.

[37] Y. R. Chen, A. Rezapour, W. G. Tzeng, and S. C. Tsai, "RL-routing: An SDN routing algorithm based on deep reinforcement learning," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 3185–3199, 2020.

[38] R. Ding, Y. Xu, F. Gao, X. Shen, and W. Wu, "Deep reinforcement learning for router selection in network with heavy traffic," *IEEE Access*, vol. 7, pp. 37109–37120, 2019.

[39] G. Stampa, M. Arias, D. Sánchez-Charles, V. Muntés-Mulero, and A. Cabellos, "A deep-reinforcement learning approach for software-defined networking routing optimisation," *arXiv preprint arXiv:1709.07080*, 2017.

[40] A. Singh, S. Sharma, and A. Gumaste, "Using deep reinforcement learning for routing in IP networks," in *2021 International Conference on Computer Communications and Networks (ICCCN)*, pp. 1–9, 2021.

[41] Kotug, "Harbor Towage," Available: https://www.kotug.com/towage/harbour-towage/. [Accessed: Aug. 5, 2024].

[42] P. Skaisgiris, W. Simoncini, F. Barbero, A. Ahangi, and R. Mockel, "PySeidon-A Data-Driven Maritime Port Simulation Framework," in *Proceedings of the 13th International Conference on Computer Modeling and Simulation*, pp. 164–171, 2021.

[43] S. Verduijn, "Identifying the relations between and mapping the processes of the nautical service providers in the Port of Rotterdam," unpublished, 2017.

[44] IMO, "Introduction to imo," Available: https://www.imo.org/en/about/pages/default.aspx.

[45] R. Zuidwijk, "Ports and global supply chains," in *Ports and Networks*, Routledge, pp. 26–37, 2017.

[46] J. Hoffmann and S. Sirimanne, "Review of maritime transport," in *United Nations Conference on Trade and Development*, 2017.

[47] M. Lind, T. Andersen, M. Bergmann, R. T. Watson, S. Haraldson, M. Karlsson, M. Michaelides, J. Gimenez, R. Ward, N. Bjørn-Andersen, et al., "The maturity level framework for PortCDM," *Sea Traffic Manage., Norrköping, Sweden, Tech. Rep.*, vol. 13, 2018.

[48] L. Kang, Q. Meng, and K. C. Tan, "Tugboat scheduling under ship arrival and tugging process time uncertainty," *Transportation Research Part E: Logistics and Transportation Review*, vol. 144, p. 102125, 2020.

[49] M. P. M. Hendriks, D. Armbruster, M. Laumanns, E. Lefeber, and J. T. Udding, "Strategic allocation of cyclically calling vessels for multi-terminal container operators," *Flexible Services and Manufacturing Journal*, vol. 24, pp. 248–273, 2012.

[50] A. H. Gharehgozli, D. Roy, and R. De Koster, "Sea container terminals: New technologies and OR models," *Maritime Economics & Logistics*, vol. 18, pp. 103–140, 2016.

[51] L. Zhen, D. Zhuge, S. Wang, and K. Wang, "Integrated berth and yard space allocation under uncertainty," *Transportation Research Part B: Methodological*, vol. 162, pp. 1–27, 2022.

[52] S. Wang, I. Kaku, G. Y. Chen, and M. Zhu, "Research on the modeling of tugboat assignment problem in container terminal," *Advanced Materials Research*, vol. 433, pp. 1957–1961, 2012.

[53] Y. Wu, J. Xiao, B. Fan, C. Wang, Y. Zhang, and K. Yao, "Research on port tugboat scheduling optimisation based on Genetic Algorithm," in *2022 2nd International Conference on Computational Modeling, Simulation and Data Analysis (CMSDA)*, pp. 165–170, 2022.

[54] P. Yao, X. Duan, and J. Tang, "An improved gray wolf optimisation to solve the multi-objective tugboat scheduling problem," *Plos One*, vol. 19, no. 2, p. e0296966, 2024.

[55] H. Yu, "Tugboat scheduling problem in large container ports: A case study of the Singapore port," in *Journal of Physics: Conference Series*, vol. 1983, no. 1, p. 012097, 2021.

[56] S. Wang, M. Zhu, J. Zheng, and K. Zheng, "Tugboat scheduling problem based on trust-based ant colony optimisation," in *Information Computing and Applications: Third International Conference, ICICA 2012, Chengde, China, September 14-16, 2012. Proceedings*, Springer, pp. 373–380, 2012.

[57] S. Nikghadam, R. Vanga, J. Rezaei, and L. Tavasszy, "Cooperation between vessel service providers in ports: An impact analysis using simulation for the Port of Rotterdam," *Maritime Transport Research*, vol. 4, p. 100083, 2023.