ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ



ΤΜΗΜΑ ΣΤΑΤΙΣΤΙΚΗΣ ΚΑΙ ΑΣΦΑΛΙΣΤΙΚΗΣ

ΕΠΙΣΤΗΜΗΣ

Εξόρυξη δεδομένων σε Μεγάλα Δεδομένα Κίνησης

Πέτρος Μανδάλης

Διδακτορική Διατριβή

Υποβλήθηκε στο
Τμήμα Στατιστικής και Ασφαλιστικής Επιστήμης
του Πανεπιστημίου Πειραιώς

Πειραιάς
Ιούνιος 2026

UNIVERSITY OF PIRAEUS



DEPARTMENT OF STATISTICS AND INSURANCE SCIENCE

Data Mining in Big Mobility Data

Petros Mandalis

PhD Thesis

Submitted to
Department of Statistics and Insurance Science
of the University of Piraeus

Piraeus
June 2026

Στην Ειρήνη, την Εβελίνα και την Κυβέλη για την στήριξη τους, στους γονείς μου και τον αδερφό μου για την έμπνευση και καθοδήγηση.

Ευχαριστίες

Θα ήθελα να εκφράσω την ειλικρινή μου ευγνωμοσύνη προς τον επιβλέποντά μου, Καθηγητή Νίκο Πελέκη, για την πολύτιμη καθοδήγηση, την ενθάρρυνση και τις ουσιαστικές παρατηρήσεις του καθ' όλη τη διάρκεια της διδακτορικής μου πορείας. Θα ήθελα επίσης να ευχαριστήσω θερμά τον Καθηγητή Ιωάννη Θεοδωρίδη και τον Αναπληρωτή Καθηγητή Ελευθέριο Κοφίδη για την πολύτιμη υποστήριξη και καθοδήγησή τους κατά τη διάρκεια των διδακτορικών μου σπουδών. Η επιστημονική τους γνώση επηρέασε καθοριστικά την πορεία της έρευνάς μου και μου έδωσε την ελευθερία και την αυτοπεποίθηση να εξερευνήσω φιλόδοξες ιδέες.

Είμαι επίσης ευγνώμων στους συναδέλφους μου στο Data Science Lab του Πανεπιστημίου Πειραιώς για τη συνεργασία και τις δημιουργικές συζητήσεις μας, οι οποίες εμπλούτισαν ουσιαστικά την παρούσα εργασία. Ιδιαίτερες ευχαριστίες οφείλω στην Εύα Χονδροδήμα για τη συνεργασία και τη σημαντική συμβολή της κατά την εκπόνηση των ερευνητικών εργασιών που περιλαμβάνονται στη διατριβή αυτή.

Θα ήθελα ακόμη να αναγνωρίσω τη συμβολή του ερευνητικού περιβάλλοντος του Data Science Lab, το οποίο προσέφερε το κατάλληλο πλαίσιο για την ολοκλήρωση της παρούσας διατριβής.

Τέλος, θα ήθελα να εκφράσω τις πιο θερμές μου ευχαριστίες στην Ειρήνη, την Εβελίνα και την Κυβέλη, καθώς και στους γονείς μου Μιχάλη και Μαρία και τον αδελφό μου Γιάννη, για την υπομονή, την κατανόηση και τη συνεχή ενθάρρυνσή τους σε όλη αυτή τη διαδρομή.

Περίληψη

Τα δεδομένα κίνησης που παράγονται από συστήματα όπως το AIS, το GPS και το ADS-B προσφέρουν πρωτοφανείς δυνατότητες για την ανάλυση και την πρόβλεψη της κίνησης πλοίων, οχημάτων και αεροσκαφών. Ταυτόχρονα, τα δεδομένα αυτά συνοδεύονται από σημαντικές προκλήσεις, που σχετίζονται με τον μεγάλο όγκο τους, τη μη κανονική δειγματοληψία, την ετερογένειά τους και την επιχειρησιακή χρονική κρισιμότητά τους. Η παρούσα διατριβή πραγματεύεται το ευρύτερο πρόβλημα της εξόρυξης γνώσης από μεγάλα δεδομένα κίνησης, μέσα από ένα συνεκτικό σύνολο συνεισφορών που καλύπτουν την πρόβλεψη τροχιών, την πρόβλεψη συγκεντρωτικών ροών, την κλιμακώσιμη μάθηση, τη μεθοδολογική ενίσχυση της βαθιάς μάθησης και την αναπαράσταση τροχιών.

Αρχικά, η διατριβή μελετά την πρόβλεψη κίνησης σε επίπεδο μεμονωμένου κινούμενου αντικειμένου μέσω του προβλήματος της Πρόβλεψης Πορείας Πλοίων (Vessel Route Forecasting, VRF). Για τον σκοπό αυτό αναπτύσσεται ένα συστηματικό πειραματικό πλαίσιο σύγκρισης, με στόχο την αξιολόγηση στατιστικών, μεθόδων μηχανικής μάθησης και νευρωνικών προσεγγίσεων υπό ρεαλιστικές συνθήκες που προκύπτουν από δεδομένα AIS, με έμφαση στην προεπεξεργασία των δεδομένων, στον σχεδιασμό της αξιολόγησης και στη συγκρισιμότητα των αποτελεσμάτων. Στη συνέχεια, η διατριβή μεταβαίνει στη συγκεντρωτική πρόβλεψη μέσω της Πρόβλεψης Ροής Θαλάσσιας Κυκλοφορίας (Vessel Traffic Flow Forecasting, VTFF), όπου το ζητούμενο είναι η πρόβλεψη της εξέλιξης της κυκλοφορίας σε έναν χωρικό διαμερισμό. Στο πλαίσιο αυτό πραγματοποιείται μια συστηματική σύγκριση μεταξύ άμεσων προσεγγίσεων, οι οποίες προβλέπουν απευθείας μελλοντικές τιμές ροής από ιστορικές παρατηρήσεις κυκλοφορίας, και έμμεσων προσεγγίσεων, οι οποίες προβλέπουν πρώτα τροχιές πλοίων και στη συνέχεια τις συγκεντρώνουν σε μελλοντικές ροές.

Με βάση αυτή τη διάκριση, η διατριβή εισάγει το UA-VTFF, ένα ενοποιημένο πλαίσιο πρόβλεψης που συνδυάζει πληροφορία πρόβλεψης σε επίπεδο πλοίου και σε επίπεδο ροής, με στόχο τη βελτίωση της ευρωστίας και της προγνωστικής επίδοσης. Η ενοποιημένη αυτή κατεύθυνση επεκτείνεται περαιτέρω μέσω του dUA-VTFF, ενός κατανεμημένου πλαισίου βασισμένου σε αρχιτεκτονική Transformer, το οποίο σχεδιάστηκε ώστε να ανταποκρίνεται στις απαιτήσεις κλιμάκωσης μεγάλων συνόλων δεδομένων AIS και να αντιμετωπίζει την υπολογιστική εφικτότητα ως κεντρική διάσταση της πρόβλεψης κίνησης και όχι ως απλή λεπτομέρεια υλοποίησης.

Πέρα από τις ίδιες τις αρχιτεκτονικές πρόβλεψης, η διατριβή συνεισφέρει και στη μεθοδολογία της βαθιάς μάθησης μέσω του KANB, μιας οικογένειας παραμετρικών συναρτήσεων ενεργοποίησης με εκπαιδεύσιμες παραμέτρους, βασισμένων σε κυβικές καμπύλες Bézier. Η οικογένεια KANB αξιολογείται ως αρθρωτή αντικατάσταση κλασικών συναρτήσεων ενεργοποίησης σε δίκτυα πρόβλεψης.

Τέλος, η διατριβή εξετάζει την αναπαράσταση τροχιών και τη συσταδοποίηση σε μη επιβλεπόμενα περιβάλλοντα. Προτείνεται ένα πλαίσιο αναφοράς για τη διανυσματοποίηση τροχιών, τη μείωση διαστασιμότητας και τη συσταδοποίηση, το οποίο επιτρέπει τη συστηματική σύγκριση διαφορετικών οικογενειών αναπαράστασης σε πολλαπλά σύνολα δεδομένων κίνησης. Η συνεισφορά αυτή αναδεικνύει ότι η αναπαράσταση δεν αποτελεί απλώς στάδιο προεπεξεργασίας, αλλά κεντρικό παράγοντα που επηρεάζει την ποιότητα της συσταδοποίησης, τη δομή ομοιότητας και τη δυνατότητα αξιοποίησης τυποποιημένων, κλιμακώσιμων εργαλείων ανάλυσης για δεδομένα τροχιών.

Συνολικά, οι συνεισφορές της διατριβής διαμορφώνουν μια ενιαία θεώρηση της ανάλυσης δεδομένων κίνησης ως πρόβλημα αναπαράστασης, πρόβλεψης, σχεδιασμού συστημάτων και μεθοδολογικής ευρωστίας. Τα αποτελέσματα δείχνουν ότι η ακριβής και κλιμακώσιμη πρόβλεψη κίνησης απαιτεί όχι μόνο αποτελεσματικά προγνωστικά μοντέλα, αλλά και αυστηρές διαδικασίες αξιολόγησης, κατάλληλα επίπεδα

αφαίρεσης, επαναχρησιμοποιήσιμα συστατικά βαθιάς μάθησης και αναπαραστάσεις που υποστηρίζουν τόσο τη μάθηση όσο και την ανάλυση σε μεγάλη κλίμακα. Παρότι ο κύριος εμπειρικός πυρήνας της διατριβής εστιάζει κυρίως στη θαλάσσια κινητικότητα, η ευρύτερη μεθοδολογική της οπτική είναι συναφής με την εξόρυξη γνώσης από δεδομένα κίνησης σε θαλάσσια, αστικά και αεροπορικά περιβάλλοντα.

Abstract

Mobility data generated by systems such as AIS, GPS, and ADS-B provide unprecedented opportunities for analyzing and forecasting the movement of vessels, vehicles, and aircraft. At the same time, these data introduce significant challenges due to their scale, irregular sampling, heterogeneity, and operational time sensitivity. This thesis addresses the broader problem of data mining in big mobility data through a coherent set of contributions spanning trajectory forecasting, aggregate flow forecasting, scalable learning, methodological deep learning, and trajectory representation.

The thesis first studies mobility forecasting at the level of individual moving objects through Vessel Route Forecasting (VRF). A systematic experimental benchmark is developed in order to compare statistical, machine learning, and deep learning approaches under realistic AIS-derived conditions, with emphasis on data preprocessing, evaluation design, and comparability. The thesis then moves to aggregate forecasting through Vessel Traffic Flow Forecasting (VTFF), where traffic evolution is predicted over a spatial partition. In this setting, a structured comparison is carried out between direct approaches, which forecast aggregate flow values directly from historical traffic observations, and indirect approaches, which first forecast vessel trajectories and then aggregate them into future flows.

Building on this distinction, the thesis introduces UA-VTFF, a unified forecasting framework that combines vessel-level and flow-level predictive information in order to improve robustness and predictive performance. This unified direction is extended through dUA-VTFF, a distributed and Transformer-based framework designed to address the scalability requirements of large AIS datasets and to treat computational feasibility as a core dimension of mobility forecasting rather than as an implementation detail.

Beyond forecasting architectures, the thesis also contributes to methodological deep learning through KANB, a family of trainable parametric activation functions based on cubic Bézier curves. KANB is evaluated as a modular replacement for standard activation functions in forecasting networks.

Finally, the thesis addresses trajectory representation and clustering in unsupervised settings. A benchmark framework is proposed for trajectory vectorization, dimensionality reduction, and clustering, enabling a systematic comparison of representation families across multiple mobility datasets. This contribution emphasizes that representation is not merely a preprocessing step, but a central factor affecting clustering quality, similarity structure, and the use of off-the-shelf scalable analytics tools for trajectory data.

Taken together, the contributions of the thesis establish a unified view of mobility analytics as a problem of representation, prediction, systems design, and methodological robustness. The results show that accurate and scalable mobility forecasting requires not only effective predictive models, but also rigorous evaluation, appropriate abstraction levels, reusable deep-learning components, and representations that support both learning and large-scale analytics. Although the empirical core of the thesis is centered primarily on maritime mobility, its broader methodological perspective is relevant to mobility data mining across maritime, urban, and aviation domains.

Table of Contents

ABSTRACT ..... **III**

List of Publications XIII

List of Figures VIII

List of Tables X

PART I. SETTING THE SCENE **1**

CHAPTER 1: INTRODUCTION **2**

1.1 Motivation 2

1.2 Application Scenarios 2

1.3 Research Challenges 3

1.4 Research Objectives 4

1.5 Thesis Contributions 5

1.6 Thesis Organization 6

CHAPTER 2: BACKGROUND AND RELATED WORK **7**

2.1 Introduction 7

2.2 Mobility Data Mining and Big Mobility Data 7

2.3 Common Data Characteristics, Datasets, and Evaluation Metrics 10

2.3.1 Datasets Used in the Thesis 11

2.3.2 Evaluation Metrics Used in the Thesis 14

2.4 Vessel Route Forecasting (VRF) 15

2.5 Vessel Traffic Flow Forecasting (VTFF) 16

2.6 Distributed and Scalable Mobility Forecasting 20

2.7 Deep Learning Foundations and Activation Functions 21

2.8 Trajectory Representation and Clustering 22

2.9 Summary 23

PART II. VESSEL ROUTE FORECASTING **25**

CHAPTER 3: COMPARATIVE BENCHMARKING OF VESSEL ROUTE FORECASTING (VRF) UNDER REALISTIC MOBILITY DATA CONDITIONS **26**

3.1 Introduction 26

3.2 Problem Formulation 27

3.3 Methodology 27

3.4 Experimental Setup 30

3.5 Results 31

3.6 Discussion 32

3.7 SUMMARY 33

PART III. VESSEL TRAFFIC FLOW FORECASTING 35

CHAPTER 4: DIRECT VS INDIRECT APPROACHES TO VESSEL TRAFFIC FLOW FORECASTING (VTFF) 36

4.1 INTRODUCTION 36

4.2 PROBLEM FORMULATION 37

4.3 METHODOLOGY 38

4.4 EXPERIMENTAL RESULTS 40

4.4.1 Experimental setup 40

4.4.2 Results 41

4.5 DISCUSSION 44

4.6 SUMMARY 45

CHAPTER 5: A UNIFIED APPROACH TO VESSEL TRAFFIC FLOW FORECASTING (UA-VTFF) 47

5.1 INTRODUCTION 47

5.2 PROBLEM FORMULATION 47

5.3 METHODOLOGY 48

5.3.1 The enhanced VRF model 48

5.3.2 The Unified Approach for VTFF (UA-VTFF) 49

5.4 EXPERIMENTAL SETUP 52

5.5 RESULTS 52

5.6 DISCUSSION 56

5.7 SUMMARY 57

CHAPTER 6: DISTRIBUTED UNIFIED VESSEL TRAFFIC FLOW FORECASTING (DUA-VTFF) 58

6.1 INTRODUCTION 58

6.2 PROBLEM FORMULATION 58

6.3 METHODOLOGY 59

6.3.1 VRF-LSTM method 60

6.3.2 The distributed Unified Approach for VTFF (dUA-VTFF) based on Transformer Networks 61

6.4 EXPERIMENTAL SETUP 64

6.4.1 Dataset description 64

6.4.2 Experimental setup 65

6.4.3 Results 68

6.5 DISCUSSION 71

6.6 SUMMARY 73

PART IV. METHODOLOGICAL ADVANCES IN DEEP LEARNING **74**

CHAPTER 7: KANB – TRAINABLE PARAMETRIC ACTIVATION FUNCTIONS **75**

7.1 INTRODUCTION 75

7.2 PROBLEM FORMULATION 75

7.3 METHODOLOGY 77

7.4 EXPERIMENTAL SETUP 78

7.5 RESULTS 80

7.6 DISCUSSION 84

7.7 SUMMARY 85

PART V. TRAJECTORY REPRESENTATION AND CLUSTERING **86**

CHAPTER 8: A BENCHMARK FRAMEWORK FOR TRAJECTORY VECTORIZATION IN UNSUPERVISED SETTINGS **87**

8.1 INTRODUCTION 87

8.2 PROBLEM FORMULATION 88

8.3 METHODOLOGY 89

8.3.1 GEOLET for Unsupervised Tasks 90

8.3.2. Dictionary-based vectorization 91

8.3.3 Distributed Clustering Methods 93

8.4 EXPERIMENTAL SETUP 93

8.4.1 Hyperparameters 94

8.4.2 Datasets 94

8.5 RESULTS 96

8.6 DISCUSSION 101

8.7 SUMMARY 102

PART VI CONSOLIDATION AND OUTLOOK **104**

CHAPTER 9: CONCLUSION **105**

CHAPTER 10: FUTURE WORK **108**

List of Figures

Figure 1. Summary of the conceptual structure of the background review 10

Figure 2. Overview of: (a) Aegean–Cyclades dataset, (b) Brest dataset 12

Figure 3. Porto Taxi Dataset 12

Figure 4. London air traffic dataset 13

Figure 5. Mobility forecasting paradigms addressed in the thesis 19

Figure 6. Experimental workflow for the comparative evaluation of vessel route forecasting methods .. 30

Figure 7. Boxplots for the FDE (in meters), on the Aegean-Cyclades in the testing set, the two UTM coordinates separately 32

Figure 8. Overview of a spatiotemporal grid of 4 x 4 space and 5 frames in time. Coloured lines represent four different trajectories. 38

Figure 9. Overview of the traffic flow for the whole period of November, 2018 for the Aegean-Cyclades dataset, within a spatial grid of G = 10km. Darker color indicates higher traffic flow. 41

Figure 10. Actual vs. predicted vessel traffic flow produced by the VTFF with XgBoost in the busiest grid cell for the training set 43

Figure 11. Actual vs. predicted vessel traffic flow produced by the VTFF with XgBoost in the busiest grid cell for validation set and testing set (zoomed graph) 43

Figure 12. Proposed VRF-LSTM network architecture, with one input layer, one LSTM layer, one fully-connected layer, and one output layer. 49

Figure 13. Overview of the UA-VTFF methodology 50

Figure 14. Example of 7x7 neighbor-surrounding traffic cells of a spatial grid. 51

Figure 15. Prediction performance in training and testing sets for grid cell id 19837 of the: (a) XgBoost-based direct VTFF, and (b) XgBoost-based UA-VTFF 54

Figure 16. Prediction performance in training and testing sets for grid cell id 19653 of the: (a) XgBoost-based direct VTFF, and (b) XgBoost-based UA-VTFF 55

Figure 17. Overview of the dUA-VTFF methodology. 60

Figure 18. Network architecture, with an encoder layer, a Self-Attention Mechanism, Feed-forward NNs, and a Decoder with a Linear Layer 63

Figure 19. dUA-VTFF efficiency experiments in the Aegean-Cyclades dataset: (a) Scalability, (b) Increased data size 71

Figure 20. Overview of the proposed framework 90

Figure 21. Strategies to transform trajectories into symbolic sequences. 92

Figure 22. A sample of the two Shape-based datasets: Straight-Right dataset and Yao dataset. 95

Figure 23. Comparison of RTA across three settings 97

Figure 24. Two dimensional PCA projections of the representation spaces produced by TIF, GEOLET and RoSITA across SR, FS datasets. Colors indicate ground-truth clusters. 100

List of Tables

Table 1. Thesis methodological landscape 9

Table 2. Main datasets used in the thesis 14

Table 3. Main evaluation metrics used in the thesis 14

Table 4. Main shortcomings identified in the reviewed literature and how the thesis addresses them 24

Table 5. Main model families compared in the vessel route forecasting benchmark 29

Table 6. Prediction results for Δt up to 30 min and r up to 6 transitions (Unit: meters) 32

Table 7. Prediction Results (SMAPE) in the testing set (20 busiest grid cells), G = 10km 42

Table 8. Prediction results (SMAPE, Jaccard) for the indirect VTFF strategy in the testing set (All Grid cells). 43

Table 9. Prediction results (SMAPE) for different alternatives of the UA-VTFF method in the testing set (G=2km) 53

Table 10. Prediction results (SMAPE) for each approach (G = 2km) 53

Table 11. Prediction results (RAE) for different methods in the Aegean-Cyclades dataset 68

Table 12. Prediction results (RAE) for different methods in the Brest dataset. 68

Table 13. Statistical significance tests for the Aegean-Cyclades dataset 69

Table 14. Execution time in the Aegean-Cyclades dataset 71

Table 15. Relative Average Error (RAE) for Aegean-Cyclades dataset for each activation function for 250 epochs and different time prediction horizon. 81

Table 16. Relative Average Error (RAE) for Porto Taxi dataset for each activation function for 250 epochs and different time prediction horizon. 82

Table 17. Relative Average Error (RAE) for London air traffic dataset for each activation function for 250 epochs and different time prediction horizon. 83

Table 18. Execution time in the Aegean-Cyclades dataset including KANB activation functions. 83

Table 19. Datasets metadata 96

Table 20. Summary of experimental results on clustering quality. 98

Table 21. Summary of experimental results on clustering diversity using RTA. 99

List of Acronyms

ADE — Average Displacement Error

AIS — Automatic Identification System

ADS-B — Automatic Dependent Surveillance–Broadcast

BPTT — Backpropagation Through Time

CART — Classification and Regression Trees

CNN — Convolutional Neural Network

CPU — Central Processing Unit

DBSCAN — Density-Based Spatial Clustering of Applications with Noise

dUA-VTFF — Distributed Unified Approach for Vessel Traffic Flow Forecasting

ELU — Exponential Linear Unit

FDE — Final Displacement Error

FM — Fowlkes–Mallows (score)

GPS — Global Positioning System

GRU — Gated Recurrent Unit

KANB — Kernel Approximation with Nonlinear Bézier

KANB1/2/3/4 — Variants of the KANB activation function

LSTM — Long Short-Term Memory

ML — Machine Learning

MLlib — Apache Spark Machine Learning Library

MLP — Multi-Layer Perceptron

OSM — OpenStreetMap

PAA — Piecewise Aggregate Approximation

PCA — Principal Component Analysis

PySpark — Python API for Apache Spark

RAE — Relative Average Error

RDD — Resilient Distributed Dataset

ReLU — Rectified Linear Unit

RNN — Recurrent Neural Network

RoSITa — Rotation- and Scale-Invariant Trajectory representation

SAX — Symbolic Aggregate approXimation

SMAPE — Symmetric Mean Absolute Percentage Error

Spark — Apache Spark

SUMO — Simulation of Urban Mobility

SVM — Support Vector Machine

tf–idf — Term Frequency–Inverse Document Frequency

UA-VTFF — Unified Approach for Vessel Traffic Flow Forecasting

VRF — Vessel Route Forecasting

VRF-LSTM — Long Short-Term Memory model for Vessel Route Forecasting

VTFF — Vessel Traffic Flow Forecasting

List of Publications

The following publications form the basis of this thesis:

E. Chondrodima, P. Mandalis, N. Pelekis, and Y. Theodoridis, “Machine learning models for vessel route forecasting: An experimental comparison,” in Proceedings of the IEEE International Conference on Mobile Data Management (MDM), 2022.[57]

P. Mandalis, E. Chondrodima, Y. Kontoulis, N. Pelekis, and Y. Theodoridis, “Machine learning models for vessel traffic flow forecasting: An experimental comparison,” in Proceedings of the IEEE International Conference on Mobile Data Management (MDM), 2022, pp. 431–436. [69]

P. Mandalis, E. Chondrodima, Y. Kontoulis, N. Pelekis, and Y. Theodoridis, “Towards a unified vessel traffic flow forecasting framework,” in Proceedings of the Workshop on Big Mobility Data Analytics (BMDA), co-located with EDBT/ICDT, 2023. [93]

P. Mandalis, E. Chondrodima, Y. Kontoulis, N. Pelekis, and Y. Theodoridis, “A transformer-based method for vessel traffic flow forecasting,” GeoInformatica, vol. 29, no. 1, pp. 149–173, 2025, doi: 10.1007/s10707-024-00521-z. [84]

G. Kanakoudis, P. Mandalis and N. Pelekis, "KANB: A Family of Parametric Activation Functions With Trainable Parameters," in IEEE Access, vol. 13, pp. 146081-146087, 2025, doi: 10.1109/ACCESS.2025.3600484. [111]

C. Landi, P. Mandalis, and N. Pelekis, “A benchmark framework for trajectory vectorization in unsupervised settings,” Workshop on Big Mobility Data Analytics (BMDA), 2026. [154]

Part I. Setting the Scene

Chapter 1: Introduction

1.1 Motivation

Modern mobility systems generate massive streams of spatiotemporal data through sensing and communication infrastructures such as the Automatic Identification System (AIS), the Global Positioning System (GPS), and Automatic Dependent Surveillance–Broadcast (ADS-B). These data provide unprecedented opportunities for monitoring, modeling, and forecasting the movement of vessels, vehicles, and aircraft, thereby supporting decision making in transportation, logistics, safety, infrastructure planning, and environmental management. At the same time, they introduce substantial challenges due to their scale, irregular sampling, heterogeneity, and noise.

Within this broader context, data mining in mobility data has become increasingly important because movement traces are not useful only as records of past activity, but also as a basis for inference, prediction, and structured decision support. Anticipating future movement patterns enables more proactive and informed action than relying on current observations alone. Accurate forecasting can improve operational efficiency, support risk mitigation, optimize resource allocation, and reduce environmental impact across multiple domains.

This thesis addresses the broader problem of data mining in big mobility data through a coherent set of contributions that span multiple levels of analysis. The first level concerns individual trajectory forecasting, where the goal is to predict the future path or next positions of a moving object. The second level concerns aggregate flow forecasting, where the goal is to predict the future evolution of movement density or traffic flow over a spatial partition. Beyond forecasting, the thesis also examines trajectory representation and unsupervised structure discovery, since raw movement data are long, noisy, and difficult to analyze directly without suitable encodings. Finally, the thesis contributes to the methodology of deep learning by studying trainable activation functions as reusable components of forecasting architectures.

A central theme of the dissertation is the big-data nature of mobility analytics. Mobility datasets are not only large in volume, but also continuous, heterogeneous, and operationally time-sensitive. As a result, the methods proposed in this thesis are evaluated not only in terms of predictive effectiveness, but also with respect to scalability, computational feasibility, and methodological comparability. In this sense, the dissertation treats mobility analytics simultaneously as a prediction problem, a representation problem, and a systems problem.

Taken together, the contributions of the thesis form a coherent research progression spanning individual movement forecasting, aggregate flow forecasting, unified and distributed forecasting architectures, methodological deep-learning improvements, and benchmark-based trajectory representation analysis. Although several empirical chapters focus on maritime data, the broader methodological perspective of the thesis is relevant to mobility data mining across maritime, urban, and aviation domains.

1.2 Application Scenarios

To motivate the research questions of this thesis, we consider representative application scenarios across multiple mobility domains in which forecasting and representation learning can have immediate operational value.

Maritime mobility

Maritime transportation is a natural and important case study because AIS generates large-scale, continuously updated movement streams with substantial practical relevance. Predictive modeling in this domain can support collision-risk mitigation, traffic supervision, berth and anchorage planning, search and

rescue, anomaly triage, maritime security, and environmental response. Forecasting vessel trajectories and regional traffic flow can also assist strategic planning tasks, such as the assessment of traffic separation schemes, offshore infrastructure placement, and emissions-aware operational policies.

Urban mobility

In urban and road transportation systems, mobility forecasting supports congestion management, route guidance, public transport coordination, fleet optimization, and infrastructure planning. Predictive traffic-flow models can improve the management of dense transportation networks, while trajectory-level modeling can support demand-aware dispatching, incident anticipation, and adaptive control.

Aviation mobility

In aviation, movement prediction is relevant for air-traffic management, airport operations, delay mitigation, and risk-aware coordination. Flight trajectories and regional traffic patterns must often be analyzed under uncertainty, irregular updates, and operational constraints similar to those found in maritime and urban mobility.

Although the empirical core of this thesis is primarily centered on AIS-based maritime mobility, the methodological questions addressed here are broader. The same issues of noisy trajectories, forecasting design, representation quality, and computational scalability arise across domains. This broader perspective is reflected especially in Chapter 8, where trajectory representations and clustering strategies are benchmarked across multiple mobility datasets.

1.3 Research Challenges

Mobility forecasting in real settings is constrained by both data characteristics and operational requirements. Key challenges motivating this thesis include:

Data quality, uncertainty, and heterogeneity

Mobility streams such as AIS, GPS, and ADS-B are imperfect. Sampling is often irregular, messages may be delayed or missing, and recorded positions can contain noise or systematic bias. Moreover, practical forecasting often requires combining several sources of information, such as object metadata, map or coastline constraints, environmental context, and region-specific operational information. This creates uncertainty both in the input data and in the interpretation of model outputs. This challenge is addressed in the thesis through controlled preprocessing, comparative benchmarking under realistic mobility-data conditions, and forecasting designs that explicitly operate on noisy and irregular spatiotemporal streams.

Scale and the “Big Data” reality

Operational mobility monitoring involves large numbers of moving objects and continuous data streams. Forecasting pipelines must therefore address high volume, high velocity, heterogeneous feature spaces, and constraints on latency and throughput. In addition, data may arrive from distributed infrastructures and require preprocessing, aggregation, and learning pipelines that remain feasible as data size grows. This challenge is addressed through the scalable and distributed perspective developed throughout the thesis, especially in Chapter 6.

Representation and similarity for trajectories

Raw trajectories are long, high-dimensional, and often unsuitable for direct analysis. Effective representations must preserve geometric, temporal, and behavioral characteristics while remaining useful for

both supervised and unsupervised tasks. This challenge is especially relevant when one seeks representations that generalize across datasets and domains. This challenge is addressed in Chapter 8 through a systematic benchmark of trajectory vectorization strategies in unsupervised settings.

Forecasting paradigms and trade-offs

Mobility forecasting can be formulated at different levels of abstraction. Individual trajectory forecasting provides fine-grained and interpretable predictions, but it can be computationally demanding and sensitive to data sparsity or missing observations. Aggregate flow forecasting operates at the level of regional density or movement intensity, which often improves scalability and operational interpretability but can obscure object-level behavior. Within aggregate flow forecasting, there is an additional distinction between direct methods, which predict future flow values directly from past flow observations, and indirect methods, which first predict individual trajectories and then aggregate them into future flows. This challenge is addressed in Chapters 4–6, which examine these trade-offs and develop unified forecasting frameworks that combine the strengths of both perspectives.

Scalability and reproducibility of evaluation

Large mobility datasets make evaluation itself a methodological challenge. Sound experimental design must support scalable training and preprocessing, fair comparison among heterogeneous models, robust train-validation-test splits that avoid temporal or spatial leakage, and transparent reporting of metrics, baselines, and computational budgets. This challenge is addressed through standardized evaluation protocols, explicit experimental design, transparent reporting of results, and, in Chapter 6, scalable distributed forecasting.

Methodological robustness in deep learning

Neural forecasting models depend heavily on architectural design choices, including the form of their activation functions. Standard nonlinearities such as ReLU, Sigmoid, and Tanh are fixed a priori and may not be equally suitable across different data regimes and prediction tasks. This challenge is addressed in Chapter 7 through the introduction of trainable activation functions that adapt their shape during learning, with the aim of improving convergence and predictive performance without changing the surrounding model architecture.

These challenges motivate the structure of the thesis as a whole. Chapters 3–6 focus on forecasting under realistic data and scalability constraints, Chapter 7 addresses methodological robustness in neural forecasting, and Chapter 8 examines trajectory representation and unsupervised structure discovery in a benchmarking setting.

1.4 Research Objectives

Based on the above motivations and challenges, the thesis pursues the following objectives:

- (O1) Establish rigorous and comparable baselines for individual trajectory forecasting under realistic mobility-data conditions.
- (O2) Characterize aggregate flow forecasting paradigms, especially the distinction between direct and indirect formulations, and clarify their operational trade-offs.
- (O3) Develop unified forecasting frameworks that combine object-level and flow-level predictive signals in order to improve robustness and predictive performance.
- (O4) Ensure scalability to large mobility datasets by introducing distributed processing and learning components that keep forecasting feasible as data volume and model complexity increase.
- (O5) Advance methodological robustness in deep learning for mobility forecasting by studying trainable activation mechanisms as modular improvements to forecasting architectures.

- (O6) Provide a benchmark framework for trajectory vectorization in unsupervised settings, with emphasis on systematic evaluation and cross-domain generalization.

Across all objectives, the thesis emphasizes reproducibility and transparency through clearly defined preprocessing pipelines, task-appropriate evaluation metrics, explicit experimental protocols, and analyses that isolate the effect of key design choices.

These objectives are addressed throughout the dissertation as follows. Objective O1 is addressed in Chapter 3 through the controlled benchmark of vessel route forecasting models. Objective O2 is addressed in Chapter 4 through the comparative study of direct and indirect flow forecasting. Objectives O3 and O4 are addressed in Chapters 5 and 6 through the development of unified and distributed forecasting frameworks. Objective O5 is addressed in Chapter 7 through the study of trainable activation functions. Objective O6 is addressed in Chapter 8 through the benchmark framework for trajectory vectorization in unsupervised settings.

1.5 Thesis Contributions

The contributions of this thesis are organized around the challenges introduced in Section 1.3. In particular, Chapters 3–6 address forecasting under noisy, irregular, and large-scale mobility data conditions; Chapter 7 addresses methodological robustness in neural forecasting; and Chapter 8 addresses representation learning and unsupervised structure discovery. Together, these contributions respond to the broader problem of data mining in big mobility data through the combined lenses of prediction, representation, and scalability. More specifically, Chapters 3 and 4 address data quality and evaluation design, Chapters 5 and 6 address unified forecasting and scalability, Chapter 7 addresses methodological robustness in deep learning, and Chapter 8 addresses representation and similarity learning for mobility trajectories.

The main contributions are summarized below.

- Comparative study of individual trajectory forecasting (Chapter 3).
    - A systematic benchmark of statistical, machine learning, and neural models for vessel route forecasting, including consistent data preprocessing, controlled evaluation splits, and comparative analysis under realistic AIS characteristics.
- Comparison of direct and indirect aggregate flow forecasting (Chapter 4).
    - A structured study of two aggregate mobility forecasting paradigms, clarifying their assumptions, strengths, and limitations under realistic AIS-derived traffic-flow data.
- Unified aggregate flow forecasting framework: UA-VTFF (Chapter 5).
    - A unified forecasting framework that couples vessel-level route forecasting with flow-level modeling, aiming to improve stability and predictive quality in sparse or rapidly changing traffic regimes.
- Distributed unified forecasting framework: dUA-VTFF (Chapter 6).
    - A distributed, Transformer-based extension of the unified framework designed for large-scale AIS processing and learning, enabling experiments on substantially larger datasets and supporting the scalability dimension of the thesis.
- KANB activation functions (Chapter 7).
    - A family of trainable parametric activation functions based on cubic Bézier curves, evaluated as modular replacements for standard activations in forecasting networks.
- Benchmark framework for trajectory vectorization in unsupervised settings (Chapter 8).
    - A systematic benchmark of representation families and clustering pipelines for trajectory analytics, with emphasis on controlled evaluation and cross-domain evidence.

Beyond predictive performance, an important contribution of the thesis is the explicit treatment of scalability as a research objective rather than an implementation detail. This is reflected in the distributed processing perspective of Chapter 6, in the benchmarking philosophy adopted in several chapters, and in the broader framing of mobility analytics under real-world data volumes.

1.6 Thesis Organization

The thesis is organized into six parts:

Part I – Setting the Scene.

- Chapter 1 introduces the motivation, application scenarios, research challenges, objectives, and contributions of the thesis.
- Chapter 2 reviews the background and related work on mobility forecasting, scalable processing, deep learning methodology, and trajectory representation.

Part II – Route Forecasting.

- Chapter 3 formulates the vessel route forecasting problem and presents a comparative experimental study of statistical, machine learning, and neural approaches for individual trajectory prediction.

Part III – Traffic Flow Forecasting.

- Chapter 4 introduces aggregate traffic flow forecasting and compares direct and indirect forecasting paradigms.
- Chapter 5 presents UA-VTFF, a unified framework that combines vessel-level and flow-level forecasting.
- Chapter 6 extends this framework with distributed processing and Transformer-based modeling to address large-scale mobility forecasting.

Part IV – Methodological Advances in Deep Learning.

- Chapter 7 introduces KANB, a trainable family of activation functions, and evaluates its impact on forecasting models. Although Chapter 7 is evaluated in forecasting settings, it is presented as a separate part because its contribution is methodological and not restricted to a single forecasting task.

Part V – Representation and Clustering.

- Chapter 8 presents a benchmark framework for trajectory vectorization in unsupervised settings, showing how representation choices affect clustering quality and cross-domain generalization.

Part VI – Consolidation and Outlook.

- Chapter 9 concludes the thesis by synthesizing the main contributions and cross-cutting findings.
- Chapter 10 outlines open challenges and directions for future work.

Chapter 2: Background and Related Work

2.1 Introduction

This chapter presents the background and related work that underpin the contributions of the thesis. Since the dissertation addresses data mining in big mobility data, the chapter does not focus only on forecasting models, but also on the broader context in which such models operate: the characteristics of mobility data, the distinction between individual and aggregate prediction tasks, the challenges posed by scale and irregular sampling, the role of neural-network design choices, and the importance of trajectory representation in unsupervised analysis.

The chapter is organized so as to move from general concepts to more specific methodological threads. We first introduce mobility data mining and the notion of big mobility data, emphasizing the main data sources, the principal mobility domains, and the analytical tasks that arise from movement data. We then summarize the common data characteristics, datasets, and evaluation metrics that recur across the experimental chapters of the thesis. This common background is important because it reduces repetition and provides a unified basis for interpreting the later methodological comparisons.

Building on this foundation, the chapter reviews the main research directions that are directly connected to the thesis contributions. These include individual trajectory forecasting, aggregate flow forecasting, direct and indirect prediction paradigms, unified forecasting strategies, distributed and scalable forecasting pipelines, trainable activation functions in deep learning, and trajectory representation for clustering and unsupervised analysis. Throughout the chapter, the literature is reviewed not only descriptively but also critically, with emphasis on modeling assumptions, problem formulations, evaluation practices, and computational implications.

This perspective is especially important in the context of the present thesis. The dissertation is motivated not only by the need for predictive performance, but also by the need for robust evaluation, clearer methodological comparison, and computational feasibility under big mobility data conditions.

2.2 Mobility Data Mining and Big Mobility Data

Mobility data mining studies how movement data generated by moving objects can be transformed into useful analytical knowledge. Such data arise in several domains, including maritime traffic, urban and road transportation, aviation, logistics, and human mobility. Typical data sources include the Automatic Identification System (AIS) for vessels, Global Positioning System (GPS) traces for vehicles and mobile devices, and Automatic Dependent Surveillance–Broadcast (ADS-B) messages for aircraft. These sources continuously generate spatiotemporal observations that describe where an object is, when it was observed, and often how it is moving, for example through speed, heading, or additional contextual attributes [2], [5], [15], [18], [100], [152], [171].

A central characteristic of mobility data is that they naturally support multiple levels of analysis. At the most detailed level, one may study the trajectory of an individual object, seeking to understand or predict its future path, destination, or behavior. At a more aggregate level, one may study traffic intensity, density, or flow over regions and time intervals. Beyond prediction, mobility data mining also includes tasks such as pattern discovery, anomaly detection, trajectory clustering, similarity search, transportation-mode inference, and semantic movement analysis [143], [149], [152], [162], [171]. The present thesis is positioned at the intersection of these analytical perspectives: it studies both individual and aggregate forecasting, while also addressing trajectory representation and unsupervised structure discovery.

The phrase big mobility data refers not only to the size of modern movement datasets, but also to the broader data-management and analytical challenges they introduce. In line with the broader big-data perspective, mobility data are characterized by high volume, because large numbers of moving objects continuously emit observations; high velocity, because these observations arrive as streams and often require near-real-time processing; high variety, because the data may combine positional, temporal, kinematic, contextual, and environmental attributes from multiple sources; and high veracity challenges, because mobility observations are often noisy, incomplete, irregularly sampled, or affected by transmission and sensing limitations [18], [100], [109], [152].

These properties have direct methodological implications. First, they affect how movement data must be preprocessed, synchronized, and represented before learning can take place. Second, they influence what kind of forecasting formulation is most appropriate, since some applications require detailed object-level predictions whereas others benefit more from regional traffic estimates. Third, they make scalability a first-class concern, since successful mobility analytics must remain computationally feasible under large and continuously updated data volumes. In this sense, mobility data mining is not only a modeling problem, but also a representation problem and a systems problem.

A further important point is that mobility-data tasks differ in the level of abstraction at which the target phenomenon is defined. In individual trajectory forecasting, the unit of interest is the moving object itself, and the target is a future path, position, or sequence of positions. In aggregate flow forecasting, the unit of interest is a spatial region or cell, and the target is the future evolution of traffic volume or density over that region. These two formulations are closely related but not interchangeable. They differ in granularity, interpretability, computational cost, and operational usefulness. This distinction is central to the structure of this thesis, since Chapters 3–6 move from individual route forecasting to aggregate flow forecasting and then to unified and distributed forecasting frameworks.

Although the empirical core of the dissertation is centered primarily on maritime mobility, the broader methodological issues addressed here are not specific to one domain. Similar challenges arise in urban and aviation settings, where movement data are also irregular, noisy, large-scale, and operationally time-sensitive. For this reason, the thesis is framed more generally as a contribution to data mining in big mobility data rather than only to maritime forecasting.

At this point, it is useful to summarize the main conceptual dimensions that organize the rest of the chapter. The first is the distinction between trajectory-level and aggregate-level forecasting. The second is the distinction between direct and indirect aggregate prediction. The third is the role of scalability, deep-learning design, and representation as cross-cutting concerns. These dimensions are reflected in Table 1 and Figure 1, which serve as a roadmap for the methodological landscape reviewed in the remainder of the chapter.

Approach category	Task level	Typical methods	Strengths	Limitations	Thesis section
Pattern- and clustering-based trajectory forecasting	Individual	Trajectory clustering, route pattern mining, probabilistic path models	Interpretable; captures recurring routes	Sensitive to similarity measure and segmentation; limited scalability	2.4
Traditional machine learning for trajectory forecasting	Individual	Regression, ELM, SVM, RBF models	Simpler training; useful baselines	Depends on feature engineering; weaker long-term temporal modeling	2.4
Deep sequence models for trajectory forecasting	Individual	RNN, LSTM, GRU, encoder–decoder, attention models	Strong nonlinear sequence modeling	Often requires regularized input; higher computational cost	2.4
Statistical and classical flow forecasting	Aggregate	ARIMA, Kalman filtering, regression, SVR	Interpretable; lightweight	Limited for complex nonlinear spatiotemporal patterns	2.5
Grid-based deep flow forecasting	Aggregate	CNN, LSTM, ConvLSTM, seq2seq models	Good for structured grids	Sensitive to grid design; loses object-level detail	2.5, 2.6
Graph-based flow forecasting	Aggregate	GNN, graph convolution, graph attention, graph Transformers	Captures relational spatial structure	Requires graph design; higher complexity	2.6
Direct flow forecasting	Aggregate	Models trained directly on historical flow sequences	Efficient; directly predicts region-level traffic	Does not explicitly model individual-object behavior	2.5
Indirect flow forecasting	Aggregate via object level	Trajectory prediction followed by aggregation	Preserves finer mobility signal	More computationally demanding; pipeline complexity	2.5
Unified / hybrid forecasting	Aggregate	Hybrid, decomposition-based, combined direct/indirect models	Combines complementary signals	More complex design; scalability becomes important	2.5
Distributed and scalable forecasting	Cross-cutting	Spark pipelines, distributed workflows, scalable Transformers	Supports large-scale data processing	More systems complexity	2.6
Trainable activation approaches	Cross-cutting	Parametric and adaptive activation functions	Improves flexibility without full redesign	Extra trainable parameters; gains vary by task	2.7
Trajectory vectorization for clustering	Representation	Feature-based, shapelet, dictionary, matrix, embedding methods	Supports clustering and unsupervised analysis	Strong dependence on representation choice	2.8

Table 1. Thesis methodological landscape

Table 1 summarizes the methodological landscape reviewed in this chapter. The taxonomy emphasizes three central dimensions of the thesis: the distinction between individual and aggregate forecasting, the contrast between direct and indirect forecasting paradigms, and the role of scalability and representation as cross-cutting concerns.

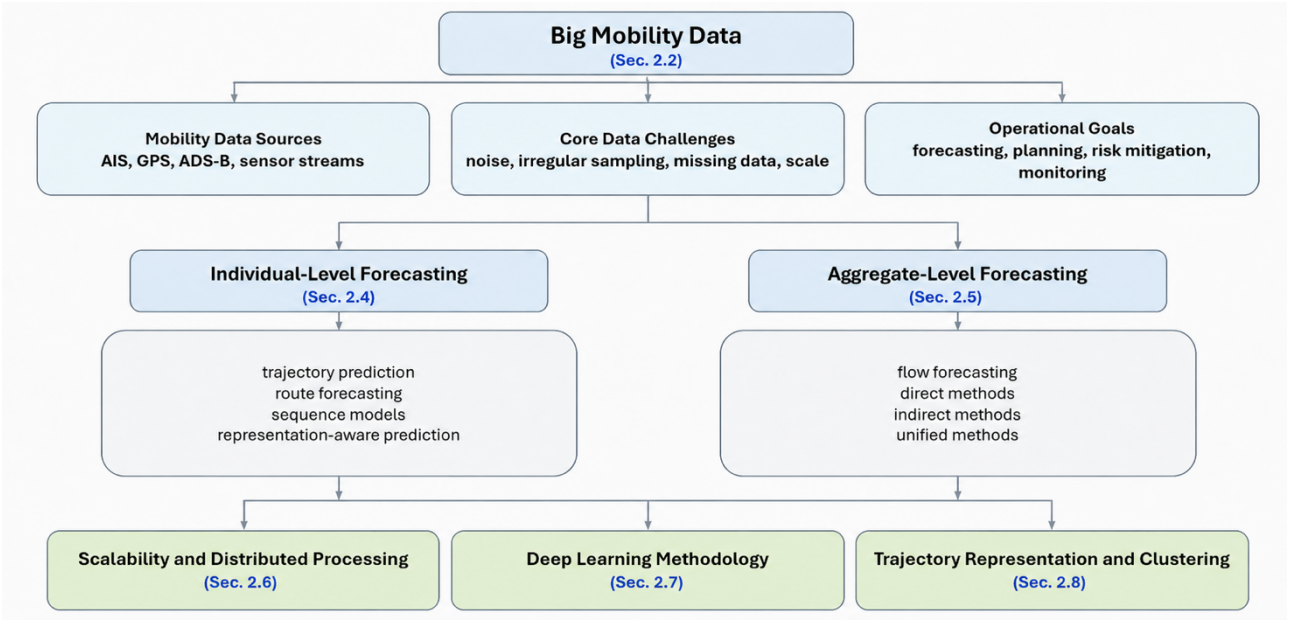


Figure 1. Summary of the conceptual structure of the background review

Figure 1 summarizes the conceptual structure of the background reviewed in this chapter. The figure highlights the relationship between mobility data characteristics, forecasting tasks at different levels of abstraction, and the cross-cutting methodological concerns of scalability, deep learning design, and trajectory representation.

2.3 Common Data Characteristics, Datasets, and Evaluation Metrics

Before reviewing the main methodological families in more detail, it is useful to summarize the data characteristics, experimental datasets, and evaluation metrics that recur throughout the thesis. This common background provides a shared point of reference for the experimental chapters and reduces the need to repeat the same definitions in later sections.

A first recurring characteristic of mobility data is irregular sampling. In practice, movement observations are rarely recorded at perfectly constant intervals. AIS, GPS, and ADS-B streams may contain gaps, variable reporting frequency, delayed messages, duplicated updates, or local bursts of dense observations. As a result, forecasting models often operate on sequences whose temporal spacing is uneven. This is an important issue because many machine learning and deep learning methods are easier to apply when inputs are regularly sampled, which often leads researchers to introduce interpolation, resampling, or temporal aggregation during preprocessing [2], [16], [18]. While these operations can simplify modeling, they may also alter the assumptions implicit in the original data-generating process.

A second recurring property is heterogeneity. Mobility datasets may include spatial coordinates, timestamps, speed, heading, route identifiers, object metadata, region-level counts, contextual signals, or externally derived features. Furthermore, the same analytical task can be represented at different levels of abstraction: as a sequence of individual positions, as a grid-based flow tensor, as a symbolic representation, or as an embedding in a learned feature space. This heterogeneity is one reason why the thesis examines not only forecasting models, but also representation choices and computational design.

A third recurring property is scale. In realistic mobility settings, datasets may include many objects, long observation periods, and large numbers of spatiotemporal records. This scale affects both learning and evaluation. It influences the choice of data structures, the feasibility of repeated training and hyperparameter

tuning, and the need for scalable processing frameworks such as distributed pipelines. Therefore, scalability is treated in this thesis as an essential methodological dimension rather than an implementation detail.

2.3.1 Datasets Used in the Thesis

The empirical chapters of this thesis rely on a set of datasets selected to cover different mobility-analysis tasks, levels of abstraction, and mobility domains. The forecasting chapters are centered primarily on AIS-based maritime datasets, while the later methodological chapters broaden the empirical scope by including additional mobility datasets for cross-domain evaluation. Centralizing the dataset descriptions in this chapter reduces repetition and provides a common reference point for the reader. The subsequent chapters therefore focus only on the experiment-specific use of the datasets, such as forecasting settings, train-validation-test splits, spatial discretization choices, and preprocessing decisions.

The two core datasets used throughout the forecasting part of the thesis are the Aegean-Cyclades dataset and the Brest dataset. Both are based on real AIS observations and therefore reflect the practical characteristics of maritime mobility data, including irregular sampling, noisy position streams, heterogeneous traffic density, and route variability. Their use is important because the thesis aims to evaluate forecasting methods under realistic mobility-data conditions rather than under simplified or artificially regularized settings.

The Aegean-Cyclades dataset in Fig. 2a, provided by MarineTraffic.com, includes a total of 1,720,368 records of location-related information derived from AIS messages transmitted by vessels sailing in the Aegean Sea, Greece, during November 2018. More specifically, it corresponds to AIS activity transmitted by 2,645 distinct vessels of various types and covers a spatial window bounded by latitude in [36.2, 38.0] and longitude in [23.2, 25.8], as illustrated in Figure 2(a). This area was selected because it includes complex and heterogeneous maritime activity, ranging from cruise and passenger traffic to fishing vessels, and encompasses routes among several Greek ports, including Piraeus, Santorini, Serifos, Milos, Sifnos, Paros, Syros, Naxos, Ios, and Hydra. Before preprocessing, the sampling interval in the dataset ranges from less than 1 second to several days, with a mean value of approximately 20 minutes and a median value of approximately 2.5 minutes. The number of messages transmitted per vessel also varies substantially, ranging from only a few observations to a maximum of 12,764 messages for a single vessel, with a median value of approximately 345 messages. These characteristics confirm that preprocessing is an essential step for machine-learning-based forecasting. After cleansing and trajectory segmentation using a maximum time gap of 30 minutes between consecutive points, the resulting dataset contains 1,156,879 records, 8,139 trajectories, and 2,344 distinct vessels, with trajectory lengths ranging from 10 to 975 points.

The Brest dataset in Fig. 2b, introduced in [43], includes a total of 18,657,858 AIS records of location-related information collected by the AIS receiver located in Brest, France, over the period from October 1, 2015 to March 31, 2016. The AIS activity was transmitted by 5,041 distinct vessels of various types and covers a sea area bounded by latitude in [-10.0, 0.0] and longitude in [45.0, 51.0], as illustrated in Figure 2(b). More information about the original dataset is provided in [43]. After combining the available dynamic and static information [43], removing invalid MMSI entries [44], and applying the preprocessing procedure adopted in this thesis, the resulting dataset contains 6,038,793 records, 20,455 trajectories, and 1,477 distinct vessels, with trajectory lengths ranging from 10 to 1,000 points. The inclusion of Brest is important because it provides a second maritime environment with different traffic characteristics, thus supporting a more robust and comparative evaluation of forecasting methods.

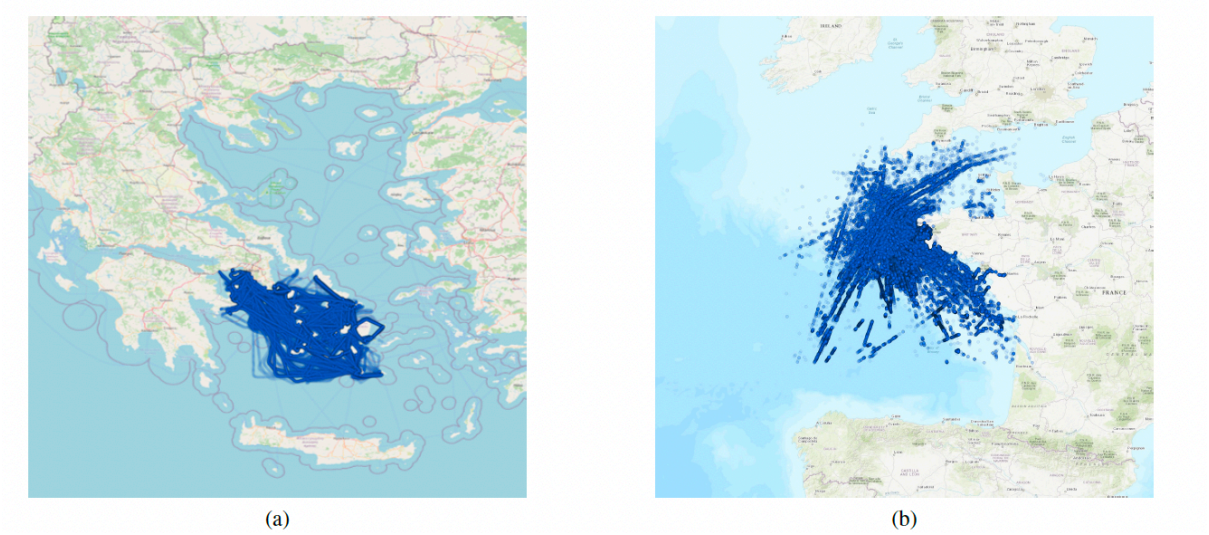


Figure 2. Overview of: (a) Aegean–Cyclades dataset, (b) Brest dataset

Beyond the maritime forecasting chapters, the thesis also employs datasets from other mobility domains in order to support broader methodological evaluation. In Chapter 7, the Porto Taxi dataset is used as an urban-mobility forecasting setting, while a London air traffic dataset is used as an aviation-mobility setting.

The Porto Taxi dataset, illustrated in Figure 3 provides a year-long record of taxi trajectories in Porto, Portugal (01/07/2013–30/06/2014), covering 442 taxis. After discretization into 551 grid cells, traffic flow values range from 0 to 67,727 vehicles per cell, with a median of 56.0 and a mean of 1993.92 vehicles. Traffic distribution is highly skewed: only 235 cells exceed 100 vehicles across the observation period, while 192 cells remain below 10 vehicles. The top 10 cells contain, on average, more than 3.43 vehicles per 5-minute interval and exceed 29,656 vehicles overall. Experiments are performed in high-traffic regions to stress-test the forecasting models under dense mobility regimes.

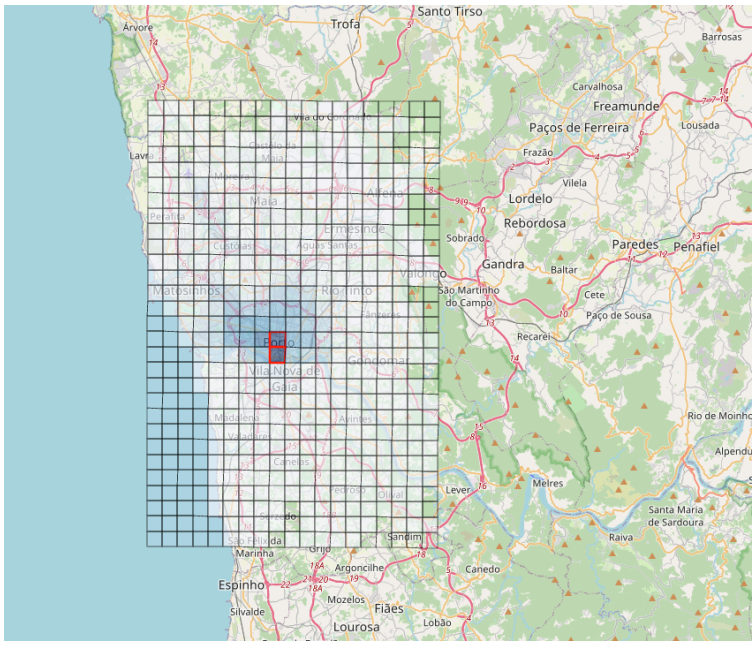


Figure 3. Porto Taxi Dataset

The London Air Traffic dataset (20 km grid), illustrated in Figure 4, contains 5,042 flights observed in the region (48.83139–54.76410°N, −5.80178–4.05083°E). A total of 1,120 grid cells are constructed, with flow values ranging from 0 to 2,951 flights per cell; the median is 0.0 and the mean is 89.35 flights. Only 187 cells exceed 100 flights during the observation period, while 810 cells contain fewer than 10 flights, indicating strong spatial sparsity. The top 10 cells exceed 378.75 flights per 5-minute interval on average and include more than 1,515 flights overall. As with the other VTFF datasets, experiments focus on high-traffic regions.

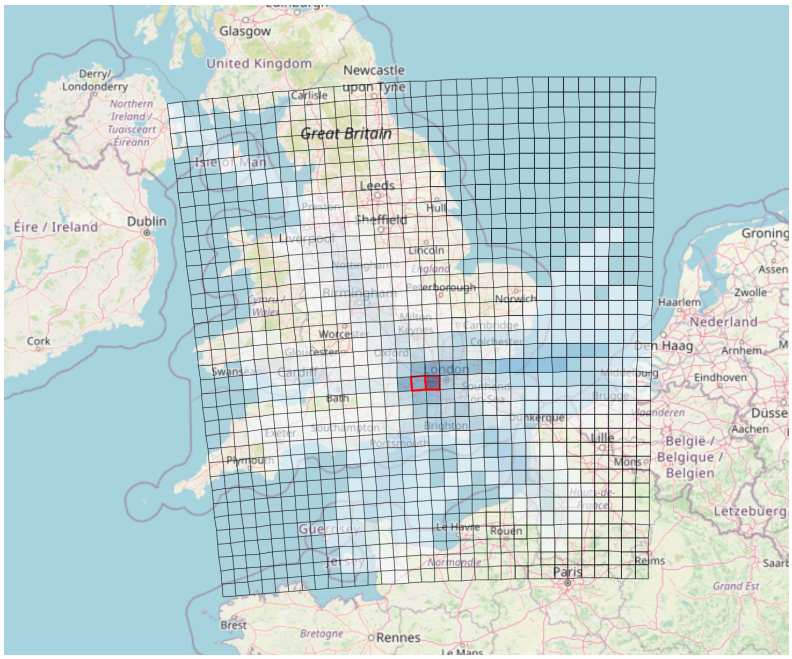


Figure 4. London air traffic dataset

Finally, Chapter 8 uses additional benchmark datasets for trajectory representation and clustering in unsupervised settings. These datasets include both controlled and more realistic trajectory collections and were selected so as to stress complementary aspects of trajectory structure, such as geometric shape, local movement patterns, behavioral similarity, and robustness to transformations. Their role in the thesis is not to support forecasting, but to enable a systematic benchmark of vectorization methods under different notions of similarity and clustering structure.

Taken together, the datasets used in the thesis support the broader methodological scope of the dissertation. Across maritime, urban, and aviation settings, they capture the central challenges addressed in this work, including forecasting under irregular sampling, scalability, representation quality, and robust evaluation.

Dataset	Domain	Main use in thesis	Data type	Used in chapters	Main characteristics
Aegean-Cyclades	Maritime	VRF, VTFF, UA-VTFF, dUA-VTFF, KANB forecasting evaluation	AIS trajectories / traffic flows	3, 4, 5, 6, 7	Real AIS data, irregular sampling, regional maritime traffic
Brest	Maritime	VRF and scalable forecasting comparison	AIS trajectories / traffic flows	3, 6	Real AIS data, different maritime traffic regime
Porto Taxi	Urban mobility	KANB mobility forecasting evaluation	GPS trajectories	7	Urban trajectory data, non-maritime mobility setting
London air traffic	Aviation	KANB mobility forecasting evaluation	Flight trajectories	7	Air-traffic trajectory forecasting setting

Chapter 8 benchmark datasets	Cross-domain	Representation and clustering benchmark	Trajectories	8	Multiple datasets emphasizing route shape, behavior, and cross-domain variation

Table 2. Main datasets used in the thesis

Table 2 centralizes the principal datasets used in the dissertation and indicates how they relate to the corresponding chapter contributions.

2.3.2 Evaluation Metrics Used in the Thesis

Because the thesis includes several related but distinct analytical tasks, different evaluation metrics are used depending on the prediction target and the level of abstraction. Centralizing these metrics here improves consistency across chapters and allows later experimental sections to focus on the interpretation of results, as described in Table 3. In the following chapters, only task-specific reminders are provided where needed.

For trajectory forecasting, the thesis uses Average Displacement Error (ADE) and Final Displacement Error (FDE). ADE summarizes the average deviation between predicted and observed future positions across the forecasting horizon, while FDE measures the deviation at the final predicted step. Together, these metrics distinguish overall trajectory quality from end-horizon accuracy.

For aggregate flow forecasting, the thesis primarily uses Symmetric Mean Absolute Percentage Error (SMAPE) and Relative Average Error (RAE). SMAPE provides a normalized measure of forecasting error that is suitable when flow magnitudes vary across regions or time periods. RAE compares the error of a forecasting method against a reference baseline and is therefore useful for relative performance assessment. In selected settings, the Jaccard index is also used in order to capture overlap-oriented aspects of aggregate prediction.

For unsupervised trajectory analysis, Chapter 8 uses the Fowlkes-Mallows (FM) score, the Silhouette score, and Representation Trajectory Alignment (RTA). The FM score measures agreement between clustering results and known grouping structure when ground truth is available, while the Silhouette score evaluates cluster compactness and separation. RTA is used as a complementary measure of how well the representation preserves trajectory-level structure under different vectorization and clustering settings.

Metric	Full name	Main task	Interpretation	Used in chapters
ADE	Average Displacement Error	Trajectory forecasting	Average position error over the prediction horizon	3
FDE	Final Displacement Error	Trajectory forecasting	Position error at the final forecasted step	3
SMAPE	Symmetric Mean Absolute Percentage Error	Flow forecasting	Scale-normalized aggregate forecasting error	4, 5
RAE	Relative Average Error	Forecasting comparison	Error relative to a baseline	6, 7
Jaccard	Jaccard index	Aggregate prediction	Overlap-based agreement measure	4
FM	Fowlkes-Mallows score	Clustering	Agreement with known grouping structure	8
Silhouette	Silhouette score	Clustering	Cluster compactness and separation	8
RTA	Representation Trajectory Alignment	Representation / clustering	Preservation of meaningful structure under representation choices	8

Table 3. Main evaluation metrics used in the thesis

Because these metrics appear repeatedly across the dissertation, it is preferable to define them once in the background chapter and then reference them in later chapters as needed. This improves consistency and helps the reader interpret the results comparatively across different methodological settings. For this reason, the subsequent experimental chapters should provide only task-specific reminders of the metrics they use, rather than reintroducing the full definitions each time.

2.4 Vessel Route Forecasting (VRF)

Vessel Route Forecasting (VRF) concerns the prediction of future vessel positions or short future trajectories from historical movement observations. This problem has attracted significant interest because it supports a wide range of operational maritime applications, including route monitoring, traffic awareness, anomaly detection, collision-risk mitigation, and local decision support [2], [5], [18]. At the same time, it introduces many of the methodological difficulties that recur throughout this thesis, such as noisy observations, irregular sampling, sparse histories, and strong spatiotemporal dependence.

A useful first distinction in the VRF literature is between pattern-based and predictive modeling approaches. Pattern-based methods attempt to exploit the fact that vessel movement is often not arbitrary: routes tend to follow recurring navigational structures, traffic separation schemes, coastline constraints, port access channels, or region-specific mobility habits. Under this view, historical trajectories are organized into representative movement patterns, and future motion is predicted by associating an incomplete trajectory with one of these learned patterns.

A representative family within this category consists of clustering-based trajectory prediction methods. Petrou et al. [14], for example, built on subtrajectory clustering [19] in order to extract trajectory fragments that can be used as predictive patterns from large mobility datasets. This line of work is appealing because it can capture recurring route structures and yield interpretable results. Similarly, Pallotta et al. [20] combined stochastic motion modeling with the TREAD method [21], an incremental unsupervised learning approach based on DBSCAN, in order to predict vessel movement while also supporting the detection of low-likelihood behaviors. Such approaches are important because they emphasize behavioral regularity and route structure rather than treating prediction as a purely black-box regression problem.

However, clustering-based approaches also have important limitations. First, they are strongly dependent on the choice of similarity measure, segmentation strategy, and clustering granularity. Second, they often rely on interpolation assumptions when converting asynchronous position updates into more regular trajectory fragments. Third, they may work well in regions with recurring traffic patterns but become less stable in areas with heterogeneous vessel behavior or weakly repeated routes. Finally, their scalability to large and continuously updated datasets is not always straightforward, especially when clustering is performed repeatedly or at fine trajectory granularity.

A second major family of VRF methods approaches the problem through supervised predictive analytics, where the goal is to learn a direct mapping from recent motion history to one or more future positions. In the maritime domain, these methods often use AIS-derived kinematic features such as latitude, longitude, speed, and course, and they may also incorporate contextual features when available. Tu et al. [5] proposed Extreme Learning Machine (ELM) models to learn vessel motion patterns from AIS data and to produce multi-horizon forecasts. Zorbas et al. [7] presented a fixed-time-window neural model for sequential location prediction from streaming trajectory data, and Valsamis et al. [8] extended this type of learning-based route prediction under different data settings. These approaches illustrate the appeal of machine learning for VRF: they can leverage large historical datasets, adapt to nonlinear motion behavior, and often outperform simple extrapolation baselines.

Nevertheless, traditional supervised learning methods are also constrained by the way vessel trajectories are preprocessed. In many cases, irregularly sampled AIS observations are transformed into regularized sequences before being passed to the model. While this simplifies the learning problem, it introduces assumptions about the underlying motion between observations. As pointed out in [2], interpolation between sampled points is a common practice in moving-object forecasting, with linear interpolation being one of the most frequently used options. Yet, as reviewed in [16], gated recurrent architectures and other sequence models are often applied to time series after regularization rather than learning directly from their irregular structure. As a result, part of the predictive success may depend as much on preprocessing decisions as on the model itself.

More recent work has increasingly relied on deep neural sequence models, especially recurrent neural networks (RNNs) and their variants. This development is consistent with the broader shift in spatiotemporal forecasting toward models that can capture temporal dependence, nonlinearity, and sequence structure more effectively than traditional methods [9], [39]. In maritime applications, Wang et al. [9] proposed a bidirectional RNN-based model combined with cubic spline interpolation for vessel berthing trajectory prediction. Capobianco et al. [10] developed an encoder-decoder approach with attention for vessel trajectory prediction, emphasizing the deep learning pipeline from raw AIS records to future route estimation. Suo et al. [11] combined clustering and recurrent modeling by using DBSCAN to derive main trajectories and then applying an RNN-based model for trajectory prediction.

This recurrent-model family is attractive because it can capture nonlinear motion dependencies over time and can be adapted to multi-step forecasting. Standard LSTM networks [35], GRU-like recurrent designs, and encoder-decoder structures [36] have all shown value in related sequence tasks. Bidirectional variants [42] and training procedures based on backpropagation through time [37] have further improved modeling capacity. In maritime route prediction specifically, more recent work has also explored sequence-to-sequence formulations [58], multipoint LSTM prediction [48], and Transformer-inspired or hybrid architectures [90], [96], [97], [107]. These methods suggest that vessel motion forecasting can benefit from the same deep sequence-modeling advances that have influenced transportation and traffic analytics more broadly.

Yet, despite this methodological progress, several recurring limitations remain visible in the VRF literature. First, many studies are difficult to compare directly because they use different datasets, different prediction horizons, different spatial resolutions, and different metrics [5], [18]. Second, many methods regularize irregular AIS streams through interpolation or resampling, which may make the forecasting problem easier but also changes the data-generating assumptions [2], [16], [18]. Third, most studies prioritize predictive accuracy while paying relatively little attention to computational feasibility, model robustness across vessel-motion regimes, or the operational implications of deploying such models in large-scale maritime systems. These gaps are especially important in the context of this thesis, which is not only concerned with forecasting performance but also with comparability, scalability, and the realistic use of AIS streams.

For these reasons, the VRF literature motivates Chapter 3 in two complementary ways. On the one hand, it provides the key algorithmic families against which a meaningful benchmark should be carried out. On the other hand, it shows why such a benchmark must be performed under a consistent preprocessing and evaluation framework if one wants to draw conclusions that are useful beyond a single study or dataset.

2.5 Vessel Traffic Flow Forecasting (VTFF)

Vessel Traffic Flow Forecasting (VTFF) shifts attention from the trajectory of an individual vessel to the aggregate evolution of maritime traffic over space and time. Instead of predicting future positions for each vessel separately, the objective is to estimate future traffic intensity, density, or flow within a spatial partition, such as a grid or a set of operational zones. This formulation is especially relevant for applications such as

proactive traffic supervision, infrastructure planning, maritime safety monitoring, and regional decision support [18], [51], [53], [54]. From the perspective of this thesis, VTFF also represents an important methodological step because it moves forecasting from the object level to the aggregate level.

One of the main advantages of VTFF is that it produces outputs that are directly useful for operational management. In many settings, maritime authorities and operators are less interested in the exact future location of every vessel than in whether traffic will intensify in a given region, passage, or port area. Aggregate forecasting therefore supports planning decisions at a higher level of abstraction and may also reduce sensitivity to object-level noise. At the same time, this formulation introduces its own modeling choices, including the definition of the spatial partition, the temporal aggregation scheme, and the representation of flow itself.

A first broad family of VTFF methods consists of direct forecasting approaches. In direct VTFF, models are trained directly on historical aggregate flow sequences and predict future flow values without an explicit vessel-level prediction stage. Early studies in this area often relied on statistical and classical machine learning methods tailored to specific maritime contexts. He et al. [53] proposed an improved Kalman-based model for short-term vessel traffic flow forecasting in multi-bridge water areas. Wang et al. [51] developed a zonal ship-flow prediction model based on machine learning for the South Atlantic States region. Zhang et al. [60] introduced a particle-swarm-optimized backpropagation model for traffic flow in a designated port area, while Kim and Lee [61] proposed a neural model for caution-area traffic prediction using AIS sensor data. Zhang et al. [62] used support vector regression with genetic optimization for narrow-water-passage forecasting, and Xie and Liu [63] applied LSTM-based sequence modeling to inland-waterway vessel flow prediction.

These direct approaches show that aggregate maritime traffic contains predictive structure that can be exploited in practice. They also illustrate the appeal of direct forecasting: the models are aligned with the operational target, they often have simpler training pipelines than vessel-level forecasting systems, and they can be computationally attractive when one is interested primarily in region-level traffic conditions. However, they also reveal an important limitation in the VTFF literature: many studies are highly region-specific. Different papers use different traffic definitions, spatial units, temporal windows, and evaluation settings, making it difficult to compare findings systematically or assess how well a method would generalize across study areas.

A second major family of VTFF methods can be interpreted as indirect forecasting approaches. In indirect VTFF, one first predicts future vessel positions or trajectories and then aggregates those vessel-level predictions into future flow maps or regional counts [54], [55], [59], [69]. Under this view, aggregate traffic is derived from object-level forecasts rather than modeled directly as a time series over regions. This approach is conceptually attractive because it preserves richer mobility information: if vessel-level forecasts are accurate, the resulting aggregate flow may better reflect emerging traffic patterns and local motion structure than a purely direct approach.

The distinction between direct and indirect VTFF is central to this thesis, but it is not always explicitly foregrounded in the literature. Many studies effectively adopt one paradigm without clearly discussing the assumptions and trade-offs that follow from that choice. This distinction matters because the two paradigms are not merely technical alternatives; they reflect different views of what should be modeled directly. Direct methods learn the dynamics of aggregate traffic itself. Indirect methods treat aggregate flow as the result of individual movement and therefore depend on the quality of an upstream trajectory-prediction stage.

This trade-off has practical consequences. Direct methods often involve lower computational cost, simpler data pipelines, and outputs that are immediately aligned with monitoring and planning tasks. Indirect methods, on the other hand, can preserve finer behavioral information, since the aggregate forecast is built from predicted object-level motion. However, indirect forecasting can become computationally expensive when

many vessels must be forecast over multiple future transitions, and any errors in the route-forecasting stage propagate into the flow forecast. In this sense, direct and indirect VTFF represent different abstractions of the same mobility phenomenon.

A particularly influential formulation of VTFF relies on grid-based spatiotemporal representations, where movement is aggregated over spatial cells and time frames. This representation is important because it links maritime traffic forecasting to the broader spatiotemporal prediction literature. In this setting, the forecasting target becomes a structured signal evolving over space and time rather than a single scalar time series. Zhou et al. [54], for example, used deep learning to forecast vessel inflow and outflow in the Singapore area. Alizadeh et al. [55] employed similarity-based prediction on AIS-derived maritime traffic, while Nguyen et al. [56] used sequence-to-sequence models over spatial grids. Experimental comparisons for vessel traffic flow forecasting were later reported in [69], and more recent Transformer-based work appears in [84].

Grid-based representations are attractive because they provide a natural and operationally meaningful description of traffic density over space and time. They also enable methodological transfer from road traffic forecasting, where rasterized or sensor-based spatiotemporal data are common [50], [89], [102]. At the same time, they introduce important design decisions. Grid resolution affects sparsity, interpretability, and model difficulty; temporal framing affects responsiveness and sequence structure; and region boundaries may shape the traffic signal in ways that are not always physically neutral. Thus, the choice of discretization is not merely a preprocessing step but part of the forecasting problem itself.

Beyond the direct-indirect distinction, a more recent direction in the literature can be described as unified or hybrid forecasting. The central idea is that aggregate forecasting may benefit from combining multiple levels of predictive information rather than relying exclusively on either direct aggregate modeling or indirect vessel-level prediction. This is the methodological direction that motivates the unified forecasting contributions of this thesis. Historical aggregate flows reveal region-level temporal structure, while individual trajectory predictions may provide complementary information about where future traffic will emerge. A framework that combines both views may therefore be more robust than one based exclusively on either representation [69], [77]–[83], [93].

Several recent studies can be interpreted as moving toward this hybrid perspective. Li and Ren [77] proposed an LSTM encoder-decoder method for multistep vessel traffic flow prediction, emphasizing sequence learning at the aggregate level. Gargari et al. [78] studied long-term traffic forecasting using both neural and seasonal time-series methods in a port setting, illustrating the coexistence of classical and learning-based approaches. Dong [79] introduced a framework that combines wavelet decomposition with deep neural forecasting, while Wang et al. [80] proposed a hybrid model based on wavelet decomposition and Prophet. Xiao et al. [81] combined a Prophet decomposition stage with GRU-based prediction and multimodal contextual information, and Rong et al. [82] forecast long-term maritime traffic by combining destination prediction with trajectory prediction before aggregating the results into a probabilistic traffic view. Xu and Zhang [83] further examined vessel traffic flow prediction through correlation analysis and GRU modeling in wind-farm areas.

These works show that aggregate maritime forecasting need not be treated as a purely one-dimensional time-series problem. Useful predictive information may exist simultaneously at several levels: regional flow history, local spatial interactions, route-specific movement, decomposition components, and contextual signals. Hybrid methods attempt to exploit this richness. However, most existing approaches still remain methodologically one-sided. Some are fundamentally direct methods enriched with decomposition or auxiliary information, while others remain fundamentally indirect methods that rely on vessel-level forecasting before aggregation. What is often missing is a framework that explicitly treats direct and indirect paradigms as complementary and combines them in a single coherent design.

This gap becomes especially important at finer spatial resolutions and longer forecasting horizons. At fine granularity, aggregate flow sequences may become sparse or unstable, which can make direct prediction harder. At the same time, indirect approaches become more computationally demanding because they require forecasting many object-level futures before aggregation. Unified approaches can therefore be viewed as a way to trade off the strengths of both paradigms: the behavioral richness of route-based prediction and the structural stability of direct flow-sequence forecasting.

For the present thesis, the VTFF literature motivates Chapter 4 and Chapter 5 in complementary ways. First, it shows that aggregate maritime forecasting is practically important but methodologically fragmented across regions, problem formulations, and evaluation settings. This motivates the controlled comparison of direct and indirect paradigms in Chapter 4. Second, it shows that the direct-versus-indirect dichotomy is not the endpoint of the problem, but rather the basis for a broader synthesis. This motivates the unified forecasting framework developed in Chapter 5, where the central question is not which paradigm is universally better, but how both can be combined in a principled and operationally useful manner.

Overall, the literature on VTFF reveals a progression from region-specific direct models, to richer grid-based and deep-learning approaches, to hybrid and unified frameworks that seek to combine complementary predictive signals. This progression aligns closely with the structure of the thesis itself and explains why aggregate traffic forecasting is treated here not as a single isolated task, but as a methodological continuum ranging from direct forecasting to indirect forecasting and finally to unified prediction.

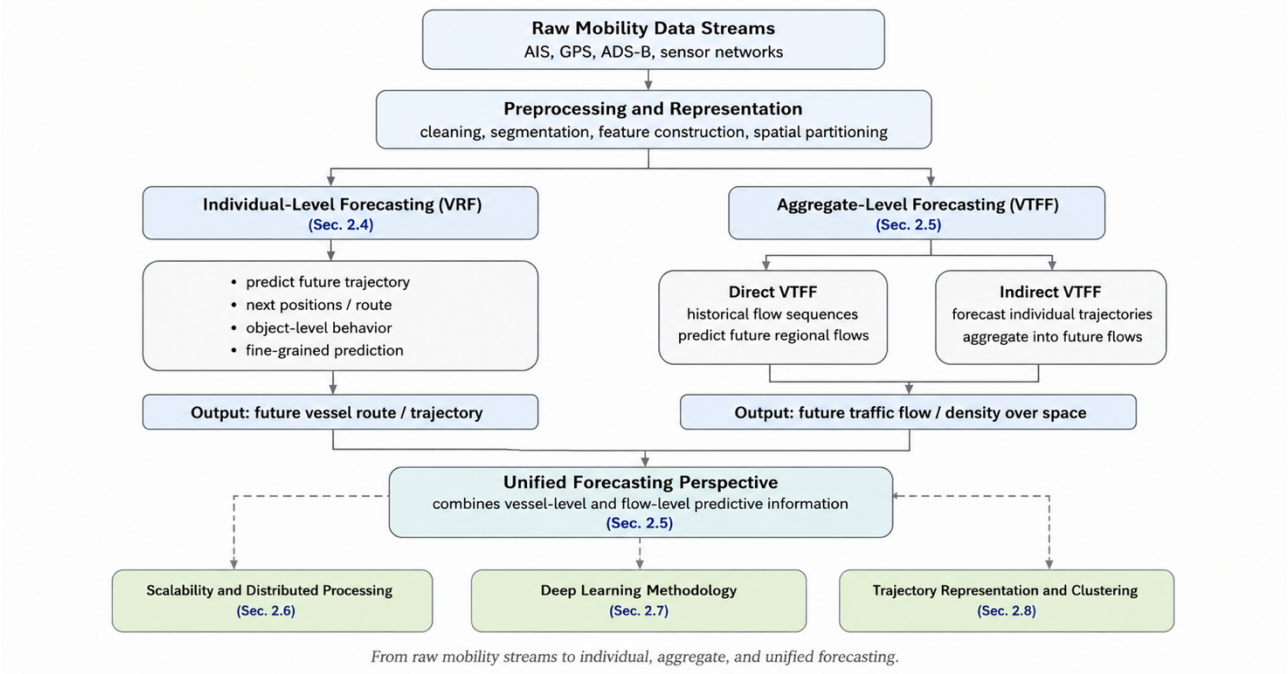


Figure 5. Mobility forecasting paradigms addressed in the thesis

Figure 5 synthesizes the main forecasting paradigms addressed in the thesis and clarifies their relationship within the broader mobility-data pipeline. After preprocessing and representation, forecasting can be formulated either at the individual-object level, through trajectory or route prediction, or at the aggregate level, through the prediction of traffic flow and density over space. In the latter case, the figure further distinguishes between direct and indirect forecasting approaches, while also highlighting the unified perspective adopted later in the thesis, where vessel-level and flow-level predictive information are combined. Finally, the figure emphasizes that scalability, deep learning methodology, and trajectory representation constitute cross-cutting methodological dimensions that influence the entire forecasting pipeline.

2.6 Distributed and Scalable Mobility Forecasting

A defining feature of modern mobility analytics is that forecasting accuracy alone is not sufficient. In real-world settings, methods must also remain computationally feasible when data volumes are large, updates are frequent, and learning pipelines must be repeatedly executed. This is especially true in AIS-based maritime monitoring, where data are generated continuously, often at high volume, and where preprocessing, feature construction, training, and inference all contribute to the overall system cost [3], [4], [14], [18], [64], [100], [109], [110], [152].

From a broader spatiotemporal-forecasting perspective, the literature commonly distinguishes among statistical, traditional machine learning, and deep learning approaches [88], [98], [102], [114]. Statistical models remain useful because they are interpretable and often lightweight, but they may fail to capture strongly nonlinear dynamics or complex spatial interactions. Traditional machine learning methods offer increased flexibility but often depend on feature engineering and may struggle when spatiotemporal dependencies become highly structured. Deep learning methods, by contrast, have become particularly influential for large spatiotemporal data because they can learn hierarchical temporal and spatial structure directly from richer inputs [88], [91], [102].

In traffic forecasting more broadly, recurrent architectures such as LSTM and GRU models have been widely used for short-term and medium-term prediction [91], [103]. More recently, there has been a strong shift toward attention-based and Transformer-style models following the influence of the Transformer architecture introduced in [95]. In the traffic domain, this has led to a variety of spatiotemporal Transformer variants, including models focused on temporal continuity and periodicity [104], locality-aware spatiotemporal attention [105], trend-adaptive architectures [106], dynamic hierarchical traffic features [99], conditional Transformer models for resilient forecasting [173], time-lag-aware Transformer variants [175], enhanced bidirectional adaptive Transformer models [172], and graph-based spatiotemporal Transformer frameworks [177]. Related work has also explored graph-convolution and graph-attention designs for traffic forecasting [115]–[118], as well as frequency-enhanced and computationally accelerated graph models [178], [179].

These developments are highly relevant because they show that the general traffic-forecasting field is increasingly treating forecasting as a joint spatial-temporal representation-learning problem. However, their relevance to maritime analytics is not automatic. Many such methods assume road-network graphs, dense sensor deployments, or urban mobility structures that do not map directly onto open-sea traffic. Maritime traffic is often more weakly constrained geometrically, more irregular temporally, and more heterogeneous in local density than urban road traffic. Therefore, importing architectures from traffic forecasting into maritime settings requires adaptation not only at the model level, but also at the level of data representation and computational design.

A second important issue is that the literature often emphasizes predictive gains without fully exposing the systems-level costs of obtaining them. End-to-end forecasting pipelines include data ingestion, cleaning, synchronization, feature generation, spatial discretization, model training, hyperparameter tuning, and inference-time deployment. In big mobility data settings, each of these steps may become a computational bottleneck. This is one reason why distributed processing frameworks and large-scale data engineering systems are increasingly relevant in mobility research [3], [4], [14], [110]. Apache Spark [110], for example, has become a standard tool for large-scale data processing and distributed machine learning workflows, and it is particularly relevant when one must repeatedly process millions of records or construct spatiotemporal datasets from raw event streams.

Another useful conceptual distinction concerns the data structures used in mobility forecasting. In the spatiotemporal data-mining literature, one may distinguish event data, trajectory data, point-reference data,

and raster data [152], [171]. Aggregate traffic-flow forecasting often operates over raster or grid-based structures because they transform raw movement observations into indexed spatiotemporal tensors that are easier to model [54], [56], [89], [102]. At the same time, graph-based structures have become important when spatial relations can be encoded more naturally as nodes and edges [50], [99], [115]–[118], [177], [178]. The choice between grid-based and graph-based representation is therefore not merely technical: it determines what spatial structure the model is allowed to learn and what kinds of computational trade-offs are required.

In the present thesis, scalability is not treated as an implementation detail but as a core research dimension. This is why the discussion of forecasting methods in Chapter 2 must extend beyond algorithmic accuracy and include issues such as large-scale processing, data representation, and distributed learning support. Chapter 6 builds directly on this background by proposing a distributed forecasting framework in which predictive performance and computational feasibility are studied together rather than in isolation.

2.7 Deep Learning Foundations and Activation Functions

Although most of the thesis contributions are organized around mobility forecasting and representation, Chapter 7 introduces a methodological deep learning contribution centered on trainable activation functions.

Activation functions are fundamental to neural networks because they define the nonlinear transformations through which information is propagated and learned. Without nonlinear activation, multilayer networks would collapse into linear mappings regardless of depth. Historically, this point goes back to foundational work on neural computation [119], and it remains central in modern deep learning [121]. In practical terms, the activation function influences convergence behavior, gradient flow, representational flexibility, training stability, and sometimes final predictive performance [124]–[127].

Classical neural forecasting architectures have often relied on standard nonlinearities such as Sigmoid, Tanh, and later ReLU and its variants [129]–[134]. These activations are popular because they are simple, differentiable in practice, and easy to integrate into gradient-based training. However, they are also fixed design choices: once selected, their shape does not adapt to the task, data regime, or network role. Over time, this limitation motivated the study of adaptive or trainable activation functions, where one or more parameters of the nonlinearity are optimized during learning.

The literature on trainable activations is broad and spans several decades. Early works studied learnable gain parameters and adaptive control-oriented neural structures [136], [137]. Later work introduced activation functions whose shape could be tuned during training [138], [139], while more recent approaches explored learnable activation families designed specifically to improve deep neural performance [140], [141]. Surveys and benchmark studies have since shown that activation choice can materially affect accuracy, convergence speed, and confidence behavior across tasks [123]–[127]. Newer activation proposals such as PolyLU [128] and broader surveys of hundreds of activation functions [126], [127] further illustrate that activation design remains an active research area rather than a solved engineering choice.

The relevance of this line of work to the present thesis is twofold. First, mobility forecasting models are often sensitive to optimization behavior because they operate on noisy, irregular, and potentially high-dimensional spatiotemporal data. Second, a methodological contribution that can improve model flexibility without replacing the surrounding architecture is especially attractive: it can, in principle, be inserted into several forecasting pipelines rather than being tied to one single task. This is precisely the conceptual role of Chapter 7.

Within this landscape, the KANB family introduced later in the thesis is positioned as a trainable parametric activation design based on cubic Bézier curves [142]. Its purpose is not to replace forecasting architectures with an entirely new modeling framework, but to provide a modular nonlinearity that adapts to

the task and can be evaluated across several predictive settings. Seen from the perspective of Chapter 2, this makes Chapter 7 a methodological branch of the thesis that complements the forecasting and representation chapters rather than standing unrelated to them.

2.8 Trajectory Representation and Clustering

Trajectory representation is a foundational issue for both supervised forecasting and unsupervised mobility analysis. The way a trajectory is encoded influences similarity computation, clustering structure, downstream predictive behavior, and computational efficiency. For that reason, a background chapter on mobility forecasting should not restrict itself to route and flow prediction models alone; it should also clarify how trajectories themselves are represented, compared, and grouped [143], [149], [152], [161], [171].

A useful conceptual point in this thesis is that unsupervised trajectory analysis can be understood as the composition of two components: a vectorization function, which maps a trajectory into a representation space, and a clustering function, which groups the resulting vectors. This perspective is important because it makes explicit that clustering quality depends not only on the clustering algorithm, but also on the representation chosen for the trajectory. In other words, poor clustering may result from an inappropriate encoding even if the clustering algorithm itself is standard.

The literature on trajectory representation spans several broad families. Feature-based approaches describe trajectories through explicit motion descriptors such as speed, heading, acceleration, turning behavior, or aggregated movement statistics [148], [150]. These methods are often attractive because they are interpretable and easy to integrate into standard machine learning pipelines. However, they depend heavily on manual feature design and may be sensitive to preprocessing decisions such as segmentation, smoothing, and resampling.

A second family consists of shapelet-based and subtrajectory-based methods, which emphasize local discriminative movement patterns rather than global handcrafted descriptors [151], [168], [170]. These methods can be more data-driven and often provide a compromise between interpretability and expressiveness. Related to this are approaches such as GEOLET and other recent subtrajectory-based representation schemes [157], which seek to capture informative local structures for trajectory analysis.

A third family includes dictionary-based or symbolic trajectory representations. These methods discretize movement into tokens or symbolic sequences, enabling the application of techniques inspired by text analytics, such as bag-of-words or tf-idf-like weighting [146], [159], [164]. Their appeal lies in their flexibility and in the possibility of using compact symbolic representations for large datasets. However, they also depend strongly on the discretization scheme and may sacrifice geometric precision for representational convenience.

A fourth family transforms trajectories into matrix- or image-like structures, allowing the use of methods originally developed for other data domains [165], [166]. More recently, deep-learning-based embeddings have become important, especially when the goal is to learn representation spaces directly from data rather than specifying them manually [160], [161], [163], [167]. Such embeddings can be powerful, but they are often tightly coupled to their training objective and may not transfer equally well across tasks or datasets.

Trajectory clustering itself has an equally rich literature. Surveys by Ahmed et al. [143], Bian et al. [145], Yuan et al. [169], and Zheng [171] show that clustering approaches differ not only in algorithmic design but also in what notion of similarity they seek to preserve: spatial proximity, temporal alignment, route shape, subtrajectory structure, semantic meaning, or behavioral similarity. Classical methods such as TRACLUS [158] illustrate the value of clustering subtrajectories rather than whole trajectories, while semantic trajectory analysis [162] expands the perspective by incorporating contextual information beyond pure movement

geometry. Recent work on shape-based mobility analysis [155] and representation learning [167] further shows that clustering quality depends deeply on the interaction between representation choice and clustering criterion.

This is precisely why trajectory representation becomes a separate contribution in the thesis rather than a minor auxiliary topic. From the perspective of the dissertation, clustering is not treated simply as an end in itself, but also as a diagnostic tool for evaluating whether a representation captures the structure one intends to preserve. A representation is useful if it yields trajectory groupings that align with a meaningful notion of similarity under a controlled experimental setting. This motivates the benchmark framework presented in Chapter 8, where multiple representation families are evaluated systematically in unsupervised settings and across multiple datasets rather than through isolated case studies.

2.9 Summary

This chapter reviewed the background and related work that underpin the dissertation, starting from the broader context of mobility data mining and big mobility data and then moving toward the main methodological directions addressed in the thesis. The review covered individual trajectory forecasting, aggregate flow forecasting, direct and indirect prediction paradigms, unified and hybrid forecasting strategies, distributed and scalable mobility analytics, trainable activation functions in deep learning, and trajectory representation for unsupervised analysis. Taken together, these topics define the methodological landscape in which the thesis contributions are positioned.

A first key observation is that mobility data mining should be understood as a broader analytical field than forecasting alone. Mobility data are generated across maritime, urban, and aviation domains, and they support multiple levels of analysis, ranging from individual movement prediction to aggregate traffic modeling and unsupervised structure discovery. This broader framing is important because it clarifies that the thesis is not limited to one application domain or one predictive task, but instead addresses a family of interrelated problems arising from large, heterogeneous, irregular, and operationally time-sensitive movement data.

A second observation is that the literature offers a wide range of forecasting models, but many studies remain difficult to compare fairly. Differences in datasets, preprocessing choices, spatial partitions, temporal horizons, and evaluation protocols often make it hard to determine whether observed performance differences reflect genuine methodological advantages or simply different experimental settings. This issue is particularly visible in both individual route forecasting and aggregate flow forecasting, and it directly motivates the benchmarking philosophy adopted in Chapters 3, 4, and 8.

A third observation is that important methodological distinctions are often underemphasized in prior work, even though they strongly affect both model design and operational interpretation. In particular, the distinction between individual and aggregate forecasting, as well as the distinction between direct and indirect aggregate prediction, is central to the structure of this thesis. The review also showed that more recent unified and hybrid approaches emerge naturally from the limitations of direct-only and indirect-only formulations, especially when fine spatial resolution, sparse traffic, or longer horizons are involved.

A fourth observation is that scalability, data representation, and deep learning design are not secondary technical details, but central components of mobility analytics under big-data conditions. Distributed processing frameworks, grid- and graph-based representations, and trainable neural components all influence not only predictive accuracy, but also computational feasibility and methodological robustness. This broader perspective is essential for understanding why the thesis moves beyond pure forecasting accuracy and treats mobility analytics as a combined problem of representation, prediction, and systems design.

These observations directly motivate the structure of the following chapters. Chapter 3 establishes a controlled comparative benchmark for individual vessel route forecasting. Chapters 4 and 5 move from

aggregate flow forecasting to the comparison of direct and indirect paradigms and then to their unified treatment. Chapter 6 extends this direction through distributed and scalable forecasting. Chapter 7 addresses methodological robustness through trainable activation functions, while Chapter 8 turns to trajectory representation and unsupervised structure discovery in a benchmark-based setting.

Overall, the chapter shows that the dissertation is positioned not within a single isolated literature thread, but at the intersection of mobility forecasting, representation learning, and scalable mobility analytics. This broader positioning is essential for addressing the central thesis problem of data mining in big mobility data and for explaining how the individual chapter contributions fit together within a coherent research progression.

Literature gap	Why it matters	Thesis response
Limited comparability across methods	Different datasets, preprocessing, and metrics make fair comparison difficult	Controlled benchmarking in Chapters 3, 4, and 8
Weak analysis of direct vs indirect flow forecasting	Important forecasting paradigms are often discussed implicitly rather than systematically	Comparative study in Chapter 4
Limited integration of trajectory- and flow-level forecasting	Separate approaches may miss complementary predictive signal	Unified frameworks in Chapters 5 and 6
Scalability often treated as secondary	Big mobility data require computational feasibility, not only predictive accuracy	Distributed forecasting framework in Chapter 6
Representation choices under-evaluated in unsupervised settings	Clustering and similarity analysis depend strongly on trajectory encoding	Benchmark framework in Chapter 8
Fixed activation functions may not facilitate model adaptation	Forecasting performance may depend on task-specific nonlinear behavior	Trainable activation study in Chapter 7

Table 4. Main shortcomings identified in the reviewed literature and how the thesis addresses them

Table 4 synthesizes the main shortcomings identified in the reviewed literature and shows how the thesis responds to them. This mapping clarifies the motivation of the subsequent chapters and makes explicit how the dissertation moves from background and related work to its own methodological contributions.

Part II. Vessel Route Forecasting

Chapter 3: Comparative Benchmarking of Vessel Route Forecasting (VRF) under Realistic Mobility Data Conditions

3.1 Introduction

The previous chapters introduced the thesis from a general mobility-data perspective, emphasizing forecasting, representation, and scalability across multiple domains. We now move to the first major empirical case study of the dissertation, namely maritime mobility, which serves as the primary application setting for the forecasting methods developed in the following chapters. This domain is particularly suitable because AIS data provide large-scale, noisy, and irregular spatiotemporal observations that reflect many of the broader challenges of big mobility data.

Maritime transport systems (MTS) are vital for global, European, and national security, finance, and trade. As a result, investments in the shipping industry continue to grow in order to improve maritime transport systems and to better monitor and understand vessel traffic. At the same time, ship-sensing technologies such as the Automatic Identification System (AIS), a self-reporting system that allows vessels to broadcast identification and voyage-related information, are generating a wealth of positioning data suitable for a wide range of applications [1].

Vessel mobility forecasting is an important component of MTS. In particular, there is growing interest in the Vessel Route Forecasting (VRF) problem, that is, the prediction of the future locations of a moving object based on its own motion history or the motion history of similar objects [2]. In this work, the informal version of the VRF problem addressed is the following: given an irregular vessel trajectory consisting of past timestamped locations, a look-ahead prediction horizon, and a look-ahead number of transitions, the goal is to predict the future trajectory of length equal to the given number of transitions at a fixed sampling rate and up to the specified prediction horizon. Accurate VRF can support improvements in travel efficiency and has a wide range of applications, including collision-risk forecasting and traffic monitoring. At the same time, VRF remains challenging due to factors such as noisy observations, irregular sampling, and stochastic influences including weather conditions.

Machine learning (ML) methods can leverage the growing volume of vessel surveillance information in order to support deeper digitalization in the shipping industry and address the VRF problem more effectively. At present, however, only a limited number of maritime monitoring systems employ ML technology in practice [3], [4]. Moving from traditional solutions to ML-based systems in the maritime domain remains challenging, and, more generally, it is not straightforward to operationalize these technologies. In the literature, several works address the route-forecasting problem either in the maritime domain [5]–[11] or in other mobility domains [12]–[15]. However, most of these works provide only limited comparative analysis, which makes it difficult to evaluate the robustness of the alternative approaches for operational use in MTS.

Due to the variable sampling rate that characterizes vessel surveillance data, most of the above works apply downsampling or resample the entire dataset through preprocessing techniques such as interpolation in order to create trajectories with a constant sampling rate. As pointed out in [2], interpolation between two consecutive sampled points is a common assumption used to approximate the continuous movement of objects, with linear interpolation being the most popular choice. However, ML methods often fail to model temporal irregularity directly and instead rely on strong assumptions about the data generation process, which may lead to poor predictive performance [16].

Different from these approaches, we address the variable sampling rate problem in vessel surveillance data by feeding the ML models with temporal and spatial information in the form of differences with respect

to previous values. The use of differences enhances neural models in handling irregular time series data [16], [17].

In summary, the main contribution of this chapter is to examine representative machine learning methods for the VRF problem and to provide comparative results based on real AIS data. The chapter is intended to serve as a reference point for understanding the relative strengths and limitations of popular forecasting approaches and to support their potential adoption in operational maritime transport systems.

3.2 Problem Formulation

This section introduces the problem formulation, presents the proposed AIS preprocessing procedure, and briefly summarizes typical solutions for VRF purposes. Also, it describes the input-output data according to the problem formulation.

Consider a maritime dataset D composed of vessel trajectories and each trajectory is a tuple $\langle o{-}id,\ S\rangle$, where $o{-}id$ is the vessel's unique identifier and S is a sequence of timestamped locations (p_i, t_i). No assumption is made about the sampling rate, i.e., $|t_{i+1} - t_i| \neq |t_i - t_{i-1}|$ in the general case.

Using D (for training purposes), the VRF problem addressed in this chapter is formulated as follows:

- Given:
  - an active trajectory, $T_a = \langle\, o - id, S_a\rangle$ where $S_a = \langle(p_1, t_1), (p_2, t_2), ..., (p_k, t_k)\rangle$,
  - a time duration (prediction horizon) Δ_t, and
  - a number of transitions r,

- Predict: the vessel's future trajectory (up to Δt) sequence of locations S'_a in r transitions equally distanced in time, i.e., $S' = \langle(p_{k+1}, t_{k+1}), (p_{k+2}, t_{k+2}), \ldots, (p_{k+r}, t_{k+r})\rangle$ where $|t_{k+1} - t_k| = |t_{k+2} - t_{k+1}| = \cdots = |t_{k+r} - t_{k+r-1}| = \Delta_t/r$

3.3 Methodology

AIS data preprocessing procedure

Due to transceiver errors AIS records contain faulty data [8]; therefore we apply typical data cleansing operations to the available AIS data, including record deduplication, noise elimination (by removing records corresponding to speed higher than 50 knots), and stationary simplification (by removing records corresponding to speed less than 1 knot). Although it may sound optional, the latter is a critical cleansing process in order for the ML model to avoid considering non-evolving parts of the trajectories. Furthermore, we did not consider vessels that turned out to have a very low number of points. In fact, we set 10 to be the minimum number of points of a trajectory of a vessel.

As already discussed, in the literature, the AIS variable sampling rate issue is often addressed by transforming variable to fixed time steps. However, this leads to higher computational load and in some cases to poor precision. Thus, in this work, instead of employing the actual AIS values, we use the differences in time and space between consecutive timestamped positions. Particularly, the time difference between k and $k{-}1$ entries in the trajectory of a vessel can be calculated as: $\Delta t_k = t_k - t_{k-1}$. Similarly, the intervals of Universal Transverse Mercator (UTM) system easting and northing are calculated by $\Delta x_k = x_k - x_{k-1}$ and $\Delta y_k = y_k - y_{k-1}$, respectively.

Furthermore, vessel trajectories, derived from AIS, are sparse. In order to produce meaningful trajectory information, in this work, original trajectories are segmented into a number of sub-trajectories when the time

interval between two consecutive points of the same trajectory exceeds a specific threshold, equal to 30 min. (this also defines the maximum prediction time interval). For technical reasons, the resulted trajectories are further segmented into a number of sub-trajectories when the length of the trajectory exceeds a specific threshold, equal to 1000 points.

Finally, we transform the processed trajectory data into a supervised learning problem by creating input-output pairs with a sliding-window procedure. Each vessel trajectory is treated as a spatial time series [7]. As discussed in [23], sequential data can be reformulated as supervised learning tasks by using past observations to predict future ones. In our case, for each trajectory, the previous observed spatiotemporal points form the input, while the next point, or next sequence of points, forms the target output. Thus, the sequential trajectory data are converted into labeled training examples suitable for supervised machine learning.

Machine learning methods

As a performance baseline, various ML methods were employed in this study to address the VRF problem. As a first step, a linear regression model [24] was employed. Also, as a representative of kernel-based algorithms we employed Support Vector Machine for regression (SVMr) [25], [26], a counterpart of Support Vector Machine (SVM) [27] for solving regression problems.

Considering tree-based methods, a classic Classification and Regression Tree (CART) algorithm [28] was employed. Focusing on Ensemble of trees techniques [29], we employed Bagging and Boosting algorithms [30], [31]. More specifically, considering algorithms of Ensemble of Trees using Bagging we selected Random Forests Trees (RFT) [31], while for the Boosting category we selected Adaptive Boosting (AdaBoost) method [32].

NNs [33] constitute one of the most powerful families of techniques in the ML arsenal. In this work, both static and dynamic networks [34] were employed. More specifically, Multi-Layer Perceptron (MLP) was used with topologies of two hidden neurons and it was trained with the classic Backpropagation algorithm [33], whereas considering the dynamic NNs category we employed two RNN [22] extended architectures, the Long Short-Term Memory (LSTM) networks [35] and the Gated Recurrent Units (GRU) [36]. The RNN-based architectures were composed of an input layer of four neurons, an RNN-based hidden layer, a fully-connected hidden layer and an output layer of two neurons. Two alternative RNN based hidden layers, GRU and LSTM were used. Details for the Backward Propagation Through Time (BPTT) algorithm, which was employed for updating the network parameters, can be found in [37]. Also, the synaptic weights were optimized according to the Adam methodology [38] on the training set.

Model Input-Output

Different ML techniques as well as different NN architectures enable different data representations to be learned [39]. All the abovementioned ML algorithms, except RNN-based models, can be applied immediately on data without time dependency, because they simply map input to output and do not consider the previous steps. More specifically, they consider all features to be independent in time, which does not apply in sequential-based problems [40].

In the case of time series related problems, the sequential information is necessary to be included through a set of delays that are added to the input, so that the data are represented at different points in time. The information relevant for making predictions need to be within a window of a number of past observations [41].

Particularly, the model input composed of the vessel's k′-th trajectory can be described by the following equation:

$$\Delta u_{k'} = (\Delta x_{k'-l}, \Delta y_{k'-l}, \Delta t_{k'-l}), \ldots, (\Delta x_{k'-1}, \Delta y_{k'-1}, \Delta t_{k'-1}), (\Delta x_{k'}, \Delta y_{k'}, \Delta t_{k'}), \Delta t_{k'+r'} \quad (1)$$

where l is the number of past observations - trajectory's points (i.e., the window size) and r' denotes the *r′-th* future transition. The corresponding model output of the r′-th future transition can be described by the following equation:

$$\Delta p_{k'} = (\Delta x_{k'+r'}, \Delta y_{k'+r'}) \quad (2)$$

We consider VRF problem as a means of a temporal multipoint location forecasting process. Thus, $\Delta t_{k'+r'}$ denotes the prediction time horizon of one point in the future. Taking into account the problem formulation described previously, the trained model needs to be executed r times, in order to give predictions for the r transitions that formulate the predicted trajectory, i.e., the model output is the following:

$$[\Delta p_{k'+1}, \ldots, \Delta p_{k'+r}] = (\Delta x_{k'+1}, \Delta y_{k'+1}), \ldots, (\Delta x_{k'+r}, \Delta y_{k'+r}) \quad (3)$$

Subsequently, the model output is transformed from differences to actual values and as a result the final model output is described by 4:

$$p_{k'+1}, \ldots, p_{k'+r} = (x_{k'+1}, y_{k'+1}), \ldots, (x_{k'+r}, y_{k'+r}) \quad (4)$$

The optimal window size l is an important parameter, as the traditional ML methods (except RNN-based architectures) can only model the data within the input window. More specifically, a small window may not provide the necessary information to the model to learn and a too large window may not be well handled by the model, i.e., the method may be unsuited for too long-term dependencies. For instance MLPs overfit on the training data when using large windows [42]. On the other hand, RNN-based architectures have the capability to dynamically incorporate past experience due to internal recurrence. Hence, if the model output of the *r′-th* future transition is described by Eq. 2, then the model input can be described by the following equation:

$$\Delta \boldsymbol{u}_{k'} = [(\Delta x_{k'}, \Delta y_{k'}, \Delta t_{k'}), \Delta t_{k'+r'}] \quad (5)$$

A central goal of this chapter is comparability. For that reason, all models are evaluated under a shared preprocessing pipeline, common forecast horizons, and the same evaluation protocol. This design reduces the confounding effect of implementation and data-handling differences and allows the comparison to focus on modeling behavior rather than on incompatible experimental settings. Table 5 summarizes the main model families considered in this chapter and highlights the main trade-offs that motivate the comparative study.

Model family	Representative methods	Main advantage	Main limitation
Statistical methods	ARIMA / baseline predictors	Interpretable and lightweight	Limited nonlinear modeling
Traditional machine learning	ELM, SVR, RBF, tree-based models	Strong baselines with moderate complexity	Depends on feature design
Neural sequence models	LSTM, GRU, encoder–decoder	Captures temporal dependencies and nonlinear motion	Higher computational cost; sensitive to data preparation

Table 5. Main model families compared in the vessel route forecasting benchmark

Figure 6 summarizes the end-to-end workflow followed in this chapter, from raw AIS data preparation to model evaluation.

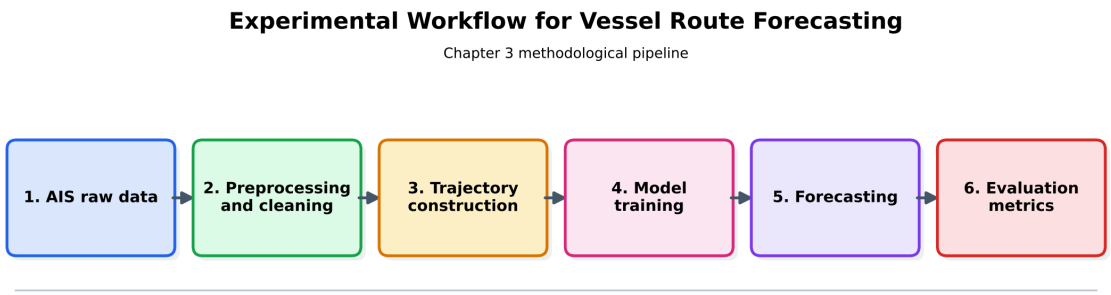


Figure 6. Experimental workflow for the comparative evaluation of vessel route forecasting methods

3.4 Experimental Setup

The datasets used in this chapter were introduced in Section 2.3.1. In particular, the empirical evaluation relies on the Aegean-Cyclades and Brest AIS datasets, which provide two complementary maritime traffic environments for benchmarking vessel route forecasting methods. Since their general characteristics have already been summarized in Chapter 2, the present section focuses only on their use within the VRF benchmark of this chapter, namely the preprocessing pipeline, the construction of forecasting instances, the experimental split design, and the comparison protocol.

The raw AIS observations are transformed into forecasting-ready trajectory sequences through a controlled preprocessing procedure. This includes the temporal ordering of observations, the removal of invalid or inconsistent records, and the construction of active trajectory segments suitable for supervised prediction. Because AIS streams are inherently irregular, the preprocessing stage is designed so as to preserve the realistic character of the data while still enabling a fair and consistent comparison across the benchmarked methods.

Each forecasting instance is constructed from an observed trajectory prefix and a corresponding future target segment. More specifically, the input consists of the past motion history of a vessel up to the current observation time, while the target consists of the future positions that must be predicted over a predefined forecasting horizon. In accordance with the problem formulation of Section 3.2, the future trajectory is represented through a fixed number of forecasted transitions over the horizon of interest. This ensures that all compared methods are evaluated on the same predictive target.

To support controlled experimentation, the data are partitioned into training, validation, and testing subsets. The training set is used for model fitting, the validation set for model selection and hyperparameter tuning, and the testing set for the final comparative evaluation. The split strategy is designed to avoid information leakage and to ensure that the reported results reflect genuine predictive generalization rather than memorization of overlapping trajectory fragments. Because fair comparison is a central goal of the chapter, the same split philosophy is applied across all benchmarked methods.

The benchmark includes representative model families from three broad categories: statistical approaches, traditional machine learning methods, and neural forecasting models. All methods are trained and evaluated under the same experimental protocol, so that differences in performance can be interpreted in methodological rather than purely experimental terms. Where required, hyperparameters are selected through the validation set, and the final reported performance is obtained only on the testing set.

The principal evaluation metrics in this chapter are Average Displacement Error (ADE) and Final Displacement Error (FDE), introduced in Section 2.3.2. In the present setting, ADE is used to assess the average spatial deviation of the predicted route over the forecasting horizon, while FDE evaluates the error at the final predicted step. Together, these metrics provide a complementary view of route-prediction quality by distinguishing overall trajectory accuracy from end-horizon accuracy.

In addition to predictive accuracy, the experimental design of this chapter emphasizes consistency and comparability. The purpose of the setup is not merely to identify the numerically strongest model on a single dataset instance, but to establish a controlled benchmark for vessel route forecasting under realistic AIS-derived conditions. This benchmarking perspective is essential for the thesis as a whole, since later chapters build on the distinction between object-level and aggregate-level forecasting and therefore require a reliable understanding of the capabilities and limitations of VRF methods.

3.5 Results

Table 6 depicts the results in terms of ADE (in meters) for a vessel's future trajectory up to Δt equal to 30 min. and r equal to 6 transitions, which results in a 5 min. sampling rate. Also, Table 6 depicts the results in terms of FDE (in meters) for vessel's future trajectory for Δt equal to 30 min. Results clearly show that the RNN-based methods outperform the static methods in terms of both ADE and FDE. This is due to the fact that the RNN-based architectures take into account the whole trajectory in contrast to the static methods, which can only model the data within the input window. Also, LSTM and GRU are able to employ large trajectories efficiently, without overfitting on the training data, or suffering from the vanishing and the exploding gradient issues. On the other hand, the static methods are unsuited for too long-term dependencies. For instance, MLPs overfit on the training data when using too large windows.

Furthermore, the SVMr method gives the worst predictions after the linear model. However, the SVMr models were trained on a subset of the available data due to the SVM's training complexity is highly dependent on the dataset size (it is known to be at least quadratic to the number of data points). Moreover, a comparison among the tree-based algorithms predictions shows that in almost all trajectory points the best solution found by the AdaBoost outperforms the RFT, followed by CART. Also, the MLP model predicts better than the CART in terms of ADE, but the CART is more accurate than the MLP in terms of the FDE in the Aegean- Cyclades dataset. Concerning the RNN-based methods, the LSTM outperforms the GRU model.

Moreover, Figures 7a and 7b depict, in the form of boxplots, the FDE in meters for the UTM eastings and northings coordinates separately. The conclusion drawn is that there do exist slight differences when comparing the error in the two coordinates but the general behaviour is more or less the same. Finally, it should be noted that we performed experiments by feeding the LSTM model with actual values instead of intervals for the Aegean-Cyclades dataset case. The produced FDE of the LSTM model by using actual values was about 8235m, i.e, about 4 times higher than the FDE produced by the LSTM with interval values, which justifies our choice to work on the differences rather than the actual coordinate values.

Data	Method	ADE per Δt in min. for r=6						FDE(30 min)
		5	10	15	20	25	30	
Aegean-Cyclades	Linear	867	1717	2569	3420	4271	5121	9371
	CART	340	889	1481	1916	2335	2796	5102
	RFT	221	654	1114	1506	1911	2377	4709
	AdaBoost	230	640	984	1374	1785	2217	4376
	SVMr	638	1335	2223	2938	3706	4310	7328
	MLP	180	735	1290	1782	2264	2765	5270
	GRU	79	195	337	511	727	977	2229
	LSTM	**76**	**184**	**317**	**481**	**684**	**920**	**2097**
Brest	Linear	1158	1788	2412	3030	3642	4312	7666
	CART	571	1091	1679	2218	2708	3247	5945
	RFT	286	641	1016	1445	1852	2226	4094
	AdaBoost	252	610	983	1387	1782	2159	4041
	SVMr	697	1388	2008	2668	3276	3828	6591
	MLP	677	1067	1482	1936	2403	2894	5344
	GRU	241	466	710	959	1215	1485	2832
	LSTM	**239**	**440**	**663**	**899**	**1146**	**1408**	**2719**

Table 6. Prediction results for Δt up to 30 min and r up to 6 transitions (Unit: meters)

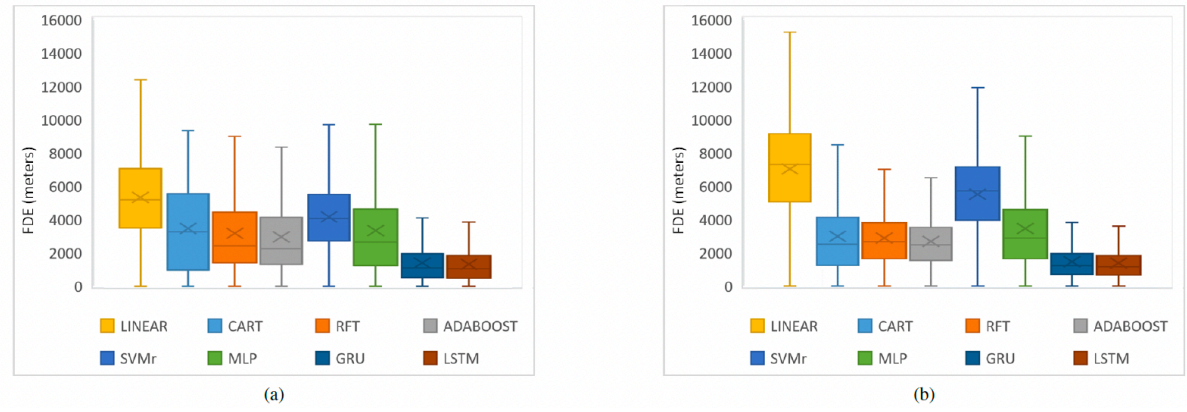


Figure 7. Boxplots for the FDE (in meters), on the Aegean-Cyclades in the testing set, the two UTM coordinates separately

3.6 Discussion

The experimental results of this chapter provide a more systematic picture of vessel route forecasting than is typically available in the literature. Rather than evaluating a single model in isolation, the chapter compares statistical, machine learning, and neural approaches under a controlled preprocessing and evaluation framework. This is important because many previous studies report strong predictive performance, yet use different datasets, horizons, spatial resolutions, or preprocessing assumptions, making direct comparison difficult. In contrast, the present benchmark aims to make performance differences interpretable in methodological rather than purely experimental terms.

A first important observation is that the relative behavior of the model families is not determined only by their nominal complexity. Simpler statistical or machine learning approaches can remain competitive under certain settings, especially when the prediction horizon is short or when the motion history contains sufficiently informative local patterns. More expressive neural models, on the other hand, become particularly valuable when the forecasting task requires capturing stronger nonlinear temporal dependencies or more complex movement structure. This confirms that, in VRF, model selection should not be viewed as a monotonic progression from simple to complex methods, but as a trade-off between representational power, robustness, and computational burden.

A second observation concerns the role of preprocessing and irregular sampling. Vessel trajectory data derived from AIS are inherently irregular, and this irregularity has a direct impact on model design. In the literature, forecasting performance is often influenced not only by the model itself but also by how the raw observations are interpolated, resampled, filtered, or transformed into model inputs. The results of this chapter reinforce the importance of evaluating models under consistent preprocessing conditions. Without such consistency, it becomes difficult to determine whether performance gains are due to a genuinely stronger forecasting method or simply to more favorable data preparation choices.

A third observation is that evaluation design is itself a methodological contribution. In real-world forecasting settings, one is not interested only in whether a model can achieve low numerical error on a specific benchmark, but also in whether its performance is stable, comparable, and meaningful under realistic mobility-data conditions. The present chapter therefore contributes to the thesis not only by reporting predictive results, but also by establishing a reproducible and comparable benchmark for individual route forecasting. This benchmarking perspective is especially important in the context of big mobility data, where methodological clarity is necessary if later chapters are to build on reliable object-level forecasting components.

From an operational perspective, the results also suggest that different VRF model families may be preferable under different priorities. If computational simplicity, interpretability, or fast deployment are critical, lighter machine learning approaches may remain attractive. If the goal is to capture more complex temporal dependencies and improve route prediction under more demanding settings, recurrent or sequence-based neural models may offer advantages. Thus, the practical value of the comparison lies not only in identifying the numerically strongest method, but also in clarifying the conditions under which each methodological family is more appropriate.

At the same time, the scope of Chapter 3 remains deliberately limited to object-level movement forecasting. While this level of analysis is essential for understanding and predicting individual vessel behavior, it does not directly address the future evolution of regional traffic intensity. In many operational settings, the main question is not where one vessel will be, but how traffic will evolve over a maritime area or spatial partition. This limitation motivates the transition to the next part of the thesis, where forecasting is reformulated at the aggregate level through Vessel Traffic Flow Forecasting (VTFF).

Overall, Chapter 3 establishes the object-level forecasting foundation of the dissertation. It identifies the main VRF model families, compares them under controlled and realistic conditions, and clarifies the role of preprocessing and evaluation design in interpreting forecasting performance. These findings provide the necessary basis for the aggregate-level studies that follow in Chapters 4–6.

3.7 Summary

This chapter examined Vessel Route Forecasting (VRF) as the first forecasting perspective of the thesis, focusing on the prediction of future vessel positions from historical movement observations. A controlled experimental benchmark was presented in order to compare representative statistical, machine learning, and neural approaches under realistic AIS-derived conditions. In this way, the chapter addressed not only

forecasting accuracy, but also the broader need for consistent preprocessing, fair evaluation, and meaningful methodological comparison.

The results showed that VRF is a nontrivial forecasting task in which model behavior depends strongly on the interaction between data characteristics, preprocessing choices, and forecasting horizon. More expressive sequence-based models can provide important advantages in capturing nonlinear movement dynamics, but simpler approaches may remain competitive in specific settings and should not be dismissed as trivial baselines. This confirms that rigorous comparison is necessary if one wishes to draw conclusions that are useful beyond a single dataset or isolated study.

An additional contribution of the chapter is methodological rather than purely empirical. By evaluating different model families within a common framework, the chapter establishes a reproducible baseline for individual trajectory forecasting in the thesis. This baseline is important because later chapters build on the distinction between object-level and aggregate-level forecasting. Without a controlled understanding of VRF performance, it would be difficult to assess the strengths and limitations of more complex frameworks that rely on route-level prediction as one component among others.

At the same time, the chapter also highlights the limitations of object-level forecasting as a complete answer to mobility analytics. Route prediction is essential when one needs fine-grained and interpretable forecasts for individual vessels, but many operational questions are formulated at a higher level of abstraction. Maritime authorities, planners, and monitoring systems are often interested in the future evolution of traffic over regions rather than in the exact path of each vessel. This shift in analytical target motivates the transition from VRF to aggregate forecasting.

For this reason, the next chapter moves from individual movement prediction to Vessel Traffic Flow Forecasting (VTFF). Chapter 4 studies how aggregate maritime traffic can be forecast over spatial partitions and, more specifically, compares the two principal paradigms of direct and indirect flow forecasting under a unified experimental perspective. In this sense, Chapter 3 provides the object-level benchmark foundation on which the aggregate forecasting analysis of the thesis is built.

More broadly, the chapter contributes to the thesis objective of establishing rigorous and comparable baselines for mobility forecasting under realistic big-data conditions. The material presented in this chapter is based on and extends the publication “Machine learning models for vessel route forecasting: An experimental comparison” [57].

Part III. Vessel Traffic Flow Forecasting

Chapter 4: Direct vs Indirect Approaches to Vessel Traffic Flow Forecasting (VTFF)

4.1 Introduction

The previous chapter studied mobility forecasting at the level of individual objects through Vessel Route Forecasting (VRF). We now move to the next level of abstraction, namely the prediction of aggregated traffic over space and time. This transition is important because many operational mobility-management tasks are not defined at the level of a single vessel, but rather at the level of regional traffic intensity, congestion, and spatial flow evolution. In this thesis, maritime mobility serves as the first major case study for this problem because AIS data provide large-scale, noisy, and irregular observations that reflect many of the broader challenges of big mobility data.

In the maritime domain, it is important to support both the safe and efficient movement of vessels. For this reason, investments in Maritime Transport Systems (MTS) continue to grow, with increasing emphasis on monitoring, understanding, and forecasting vessel traffic. Vessel Traffic Flow Forecasting (VTFF) is therefore a crucial capability for maritime navigation and traffic management. Forecasting vessel flows can support congestion mitigation, enhance maritime safety, and assist both maritime authorities and individual vessel users in route planning and operational decision making.

In the literature, traffic-flow forecasting methods can be broadly classified into parametric and non-parametric approaches [50], [51]. The former category includes time-series models that often rely on prior assumptions regarding the data-generating process, such as Autoregressive Integrated Moving Average (ARIMA) models [52]. The latter category includes data-driven methods based on patterns learned from historical data, such as neural-network models and other machine learning approaches [33], [53]. In recent years, the most promising approaches have increasingly relied on data-driven learning because they can model complex spatiotemporal structure without requiring strong parametric assumptions.

For the VTFF problem, a particularly important methodological distinction concerns the way future traffic flow is produced. In this chapter, we use the term indirect VTFF to refer to approaches that first predict future vessel trajectories or positions and then aggregate them into future traffic flow estimates. We use the term direct VTFF to refer to approaches that predict future traffic flow directly from historical flow sequences over a spatial partition. Although both paradigms appear in the literature, they are often treated separately, and their comparative strengths and weaknesses are not always made explicit.

This chapter therefore approaches the VTFF problem from both perspectives using machine learning methods and real AIS data. While several studies address vessel traffic flow forecasting with learning-based approaches [51], [53]–[55], [59]–[63], most focus on specific maritime environments such as rivers, ports, bridges, or other local areas of interest. By contrast, the aim of this chapter is to provide a broader comparative view of the problem and to examine how direct and indirect strategies behave under a common experimental setting.

In summary, the main contribution of this chapter is a systematic comparison of two major perspectives for addressing VTFF—indirect forecasting through route prediction and direct forecasting through flow-sequence prediction—using real AIS data and representative machine learning models. The chapter is intended to serve as a reference point for understanding the trade-offs between these paradigms and for supporting their future use in operational maritime transport systems.

4.2 Problem Formulation

In order to compare the two forecasting paradigms in a clear and consistent way, we formulate VTFF at the level of a spatiotemporal grid over the maritime study area. This allows both approaches to be evaluated under a common representation, while preserving the distinction between flow values produced directly from historical aggregate sequences and flow values obtained indirectly from forecasted vessel trajectories.

Consider a maritime dataset D composed of vessel trajectories and each trajectory is a sequence of timestamped locations (t_i, p_i), which consists of timestamp t and location p composed of two coordinate values, x and y, in a Cartesian projection system. The VTFF problem addressed in this work is formulated as follows:

Definition 1. Traffic Flow Forecasting:

- Given:
  - a maritime dataset of vessel trajectories over a study area,
  - a spatial partition of the study area into grid cells,
  - a temporal partition into consecutive time frames,
  - a prediction horizon Δt, and
  - a number of future temporal transitions r.

- Predict: for each grid cell and for each of the next rtime transitions up to the prediction horizon Δt, the corresponding traffic-flow value, defined as the number of vessels located within that grid cell during that future time frame.

The objective is to estimate future aggregate vessel movement over space and time under a common experimental setting, enabling comparison between direct and indirect VTFF approaches.

Under this formulation, the indirect VTFF strategy predicts future vessel positions or trajectories first and then aggregates them into future traffic-flow values over the grid. The direct VTFF strategy operates directly on historical flow sequences over the grid and predicts future traffic-flow values without an intermediate trajectory-prediction stage.

As an example, Figure 8 illustrates a spatiotemporal grid of 4 x 4 space and 5 frames in time. The four illustrated vessel trajectories evolve over time in a window of (let us suppose) 3 time frames, and the goal of the VTFF problem is to predict the expected number of vessels in each cell of the grid during the 2 future (upper in the figure) time frames. In the following subsections we present the proposed VTFF approaches.

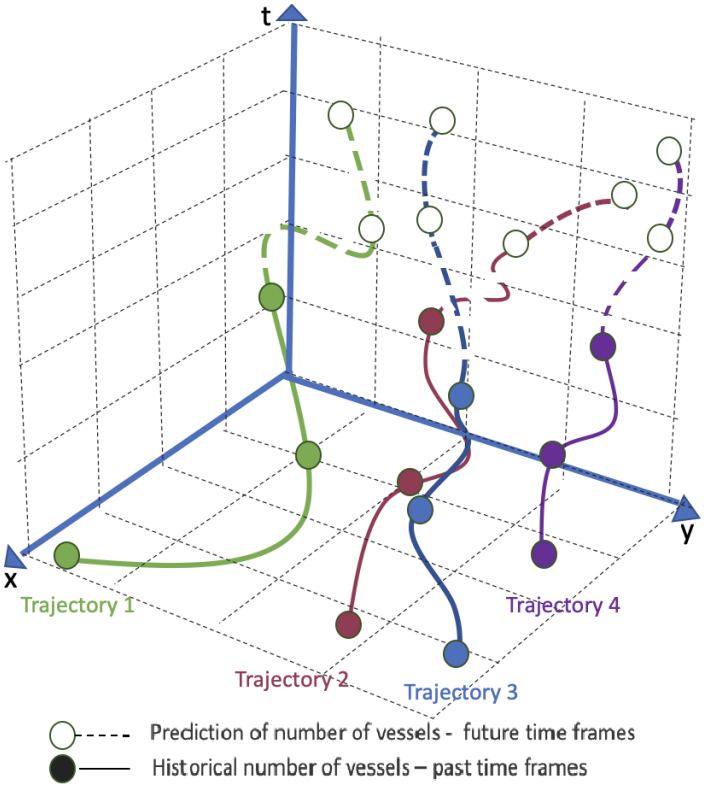


Figure 8. Overview of a spatiotemporal grid of 4 x 4 space and 5 frames in time. Coloured lines represent four different trajectories.

4.3 Methodology

The methodology of this chapter is organized around two alternative forecasting strategies. The first is the indirect VTFF strategy, in which future vessel trajectories are first forecast and then spatially aggregated into flow values. The second is the direct VTFF strategy, in which historical aggregate flow sequences are modeled directly as a spatiotemporal prediction task. Studying both strategies under the same experimental framework allows the chapter to compare their assumptions, strengths, and limitations in a more systematic way than is typically done in prior work.

A.Indirect VTFF

In this approach, the VTFF accuracy depends on the prediction performance of the underlying VRF method. In this study, in order to predict future vessels' locations we employ the algorithm presented in [9], where the trained ML model is executed r times to provide predictions for the r transitions that formulate the predicted trajectory.

In [57], some of the most popular ML methods are being investigated to address the VRF problem, including linear regression, tree-based methods, Support Vector Machine for regression (SVMr) [26], and static and dynamic NNs. Also, the experimental study in [57] proved the dominance of the LSTMs against their rivals. Thus, in this work, we use LSTMs to predict the future vessels' locations (up to Δt) given vessel trajectory coordinates.

In order to employ the algorithm in [57], a few preprocessing steps are necessary: i) stationery simplification (remove records corresponding to speed less than one knot) and insignificant trajectory elimination (remove trajectories composed of very low number of points; less than ten points), and ii) trajectory

segmentation into a number of sub-trajectories when the time interval between two consecutive points of the same trajectory exceeds a specific time threshold, equal to 30 min.

The predicted locations resulted by VRF are allocated into the spatiotemporal grid described earlier and the number of vessels located within each cell of the grid is calculated. The resulted numbers represent the volume of vessels, i.e., the traffic flow in the given area and time window.

Regarding the method evaluation and model parameter selection:

1) Method evaluation: The evaluation metrics used in this chapter were introduced in section 2.3.2. Here, they are considered specifically in the context of aggregate vessel traffic flow forecasting. In particular, SMAPE is used as the primary measure of aggregate prediction error:

$$SMAPE = \left(\frac{1}{B}\right)\sum_{b=1}^{B}\left(\frac{1}{F}\right)\sum_{t=1}^{F} 2 * \frac{|y_{b,t} - \hat{y}_{b,t}|}{|y_{b,t}| + |\hat{y}_{b,t}|} \tag{6}$$

$$Jaccard = \left(\frac{1}{B}\right)\sum_{b=1}^{B}\left(\frac{1}{F}\right)\sum_{t=1}^{F} \frac{|Y_{b,t} \cap \hat{Y}_{b,t}|}{|Y_{b,t} \cup \hat{Y}_{b,t}|} \tag{7}$$

where B and F are the total number of cells and time frames in the grid, respectively; y and ˆy represent the actual and the predicted number of vessels, respectively; Y and $\hat{Y}$ are the sets including the actual and the predicted vessels, respectively.

2) *Model parameter selection:* As far as the NN parameter selection is concerned, both the theoretical and experimental perspectives are considered. Also, an early stopping procedure [46] is applied to the validation set to prevent the NN model from overfitting, i.e. the inability of NN to predict correctly based on the data used during the training stage.

B. Direct VTFF

In this approach, the vessels' trajectories are allocated to the spatiotemporal grid to formulate the traffic flow sequence in each cell. Then we feed an ML model with these flow sequences.

The trained model can be used to predict the number of vessels in a specific cell in future time frame Δt. The information relevant for making predictions needs to be within a window of a number of past observations [41]. Particularly, the model input N is composed of a number of vessels n in each time frame t, in the b-th box within the grid of l total boxes can be described by the following equation:

$$\boldsymbol{N}^{b} = \left[n^{\mathrm{b}}_{\{\mathrm{t}-\mathrm{l}\}}, \ldots, n^{\mathrm{b}}_{\{\mathrm{t}-1\}}, n^{\mathrm{b}}_{\mathrm{t}} \right] \tag{8}$$

The corresponding model output (predicted number of vessels) in the *b*-th box grid in the future time frame t + 1 is: $n^{b}_{\{\mathrm{t}+1\}}$.

Different ML techniques enable different data representations to be learned [39]. In this study we employ XgBoost [30] and ARIMA models. XgBoost [30] is a sparsity-aware algorithm for sparse data and weighted quantile sketch for approximate tree learning, which allows the handling of large datasets with a scalable tree boosting system. Moreover, ARIMA models are able to capture a suite of different standard temporal structures in time series data. The aforementioned algorithms can be applied immediately on data because they simply map input to output.

Below the method evaluation and model parameter selection of this method is described:

1) *Method evaluation:* The traffic flow sequences for each grid cell are arranged into input and output data. For each grid cell, the initial 75% of the traffic flow sequence is used for the training purpose, whereas the remaining 25% of the traffic flow sequence, except the last three observations, is organized in the validation set. The last three observations are used for the testing purposes. Experimental results were evaluated by using SMAPE, introduced in section 2.3.2 and described by Eq.6. Due to the proposed algorithm's nature, the evaluation is performed only in the busy grid cells, i.e. regions of high traffic areas where regular navigation activities occur and are associated with a high risk of accidents [51].

2) Model parameter selection: As far as the model parameterization is concerned, several aspects of each model type were taken into consideration and adjusted with intermediate experiments before the final performance assessment. Particularly, the XgBoost considers three different types of parameters: general tree parameters, booster parameters, and learning task/miscellaneous parameters. We optimize XgBoost models according to: a) learning rate b) the minimum leaf size for pruning, c) the number of features on a node, and d) the number of regression trees. Also, the ARIMA model considers three different types of parameters: the lag order, the degree of differencing, and the order of the moving average. We optimize ARIMA models according to the above three parameters by evaluating the Partial Autocorrelation plot (PACF), the Augmented Dickey Fuller (ADF) test and the Autocorrelation plots (ACF), respectively.

4.4 Experimental Results

This section presents the chapter-specific experimental setup and the comparative results obtained for the two VTFF strategies. Since the dataset and the evaluation metrics have already been introduced in Sections 2.3.1 and 2.3.2, respectively, the focus here is on the experimental configuration that is specific to the comparison between direct and indirect aggregate forecasting.

4.4.1 Experimental setup

The two VTFF methods were implemented in Python 3. The experiments were carried out on a workstation equipped with 64 GB RAM, an Intel Core i9-9900KX CPU, and a GeForce RTX 2080 Ti GPU with 11 GB graphics memory. These implementation details are reported for completeness, since the two forecasting pipelines differ not only in their predictive logic but also in their computational structure.

For the experimental study, the Aegean-Cyclades AIS dataset introduced in Section 2.3.1 was used. In the context of the present chapter, this dataset is employed to construct a spatiotemporal grid representation over the maritime area of interest. The grid is used as the common experimental space in which both direct and indirect VTFF are evaluated. The chapter-specific preprocessing procedure includes record deduplication, by removing records with timestamps differing by less than one second, and noise elimination, by removing records corresponding to vessel speed above 50 knots. These steps are intended to support a fair comparison by reducing obvious inconsistencies in the raw AIS stream.

For the indirect VTFF strategy, the route-forecasting stage uses the LSTM-based VRF procedure described in Section 4.3.1. The resulting predicted vessel locations are then allocated to the spatiotemporal grid in order to derive future flow values. For the direct VTFF strategy, vessel trajectories are first aggregated into traffic-

flow sequences over the grid cells, and those sequences are then used as the input to XGBoost and ARIMA models. In both cases, the same spatial and temporal grid configuration is used so that the outputs remain comparable.

The evaluation metrics used in this chapter, including SMAPE and Jaccard, were introduced in Section 2.3.2. In the present chapter, SMAPE is used as the main error measure for aggregate flow prediction, while Jaccard is used in the indirect setting in order to quantify the overlap between actual and predicted vessel sets within grid cells. The chapter compares the two VTFF paradigms primarily in terms of their short-term forecasting ability under common spatial and temporal settings.

Figure 9 illustrates the traffic flow over the whole period of November 2018 for the Aegean-Cyclades dataset using a spatial grid of G=10 km. Darker color indicates higher traffic-flow intensity. This visualization provides an overview of the spatial distribution of maritime activity and motivates the distinction between busy and non-busy regions that becomes relevant in the subsequent analysis.

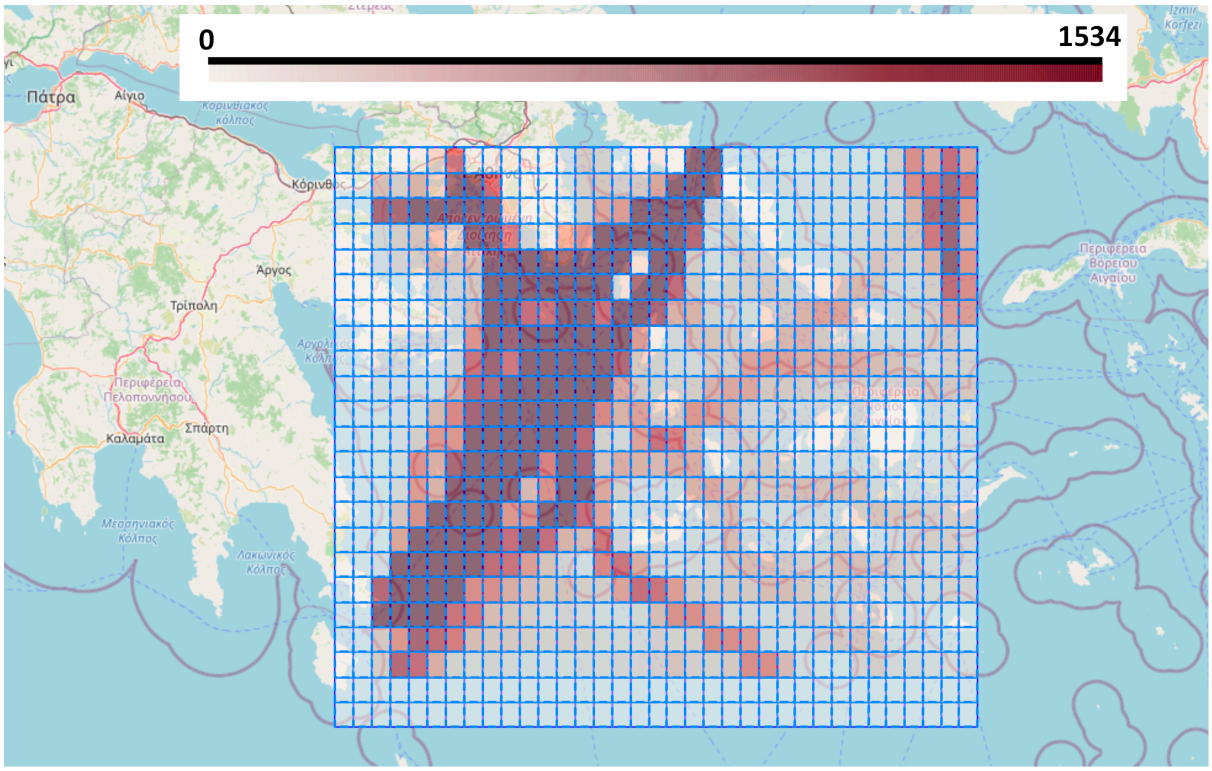


Figure 9. Overview of the traffic flow for the whole period of November, 2018 for the Aegean-Cyclades dataset, within a spatial grid of G = 10km. Darker color indicates higher traffic flow.

4.4.2 Results

As already discussed, the two alternative VTFF strategies handle differently the AIS data. Although the method evaluation for the two strategies is different, we can still compare their prediction capability in busy regions for certain time horizon.

Table 7 depicts the results of both VTFF strategies in terms of SMAPE for vessels' future flow forecasting in the 20 most busy grid cells up to Δt = 15 min. The spatiotemporal grid that was employed separates the maritime region into grid cells of G = 10 km and the time window into time frames of 5 min. Regarding the direct VTFF strategy prediction results, the XgBoost outperforms the ARIMA. More specifically, the XgBoost model predicts the future traffic flow for the next 5 min about 2.5 times better than the ARIMA model. As far

as the indirect VTFF strategy is concerned, it predicts the traffic flow up to 10 min 1.5 to 2.5 times better than VTFF with XgBoost. However, the XgBoost model performs better in the 3rd five-minute time frame.

Table 8 focuses on the indirect VTFF strategy and presents results in terms of SMAPE and average Jaccard similarity in all grid cells up to $\Delta t = 15$ min. We employed three different grids of G equal to 5 km, 10 km and 15 km, and r = 3 transitions of 5 min each. It is obvious that this method is affected by the granularity of the grid; it predicts better in larger grid cells. As far as the results in terms of the Jaccard similarity are concerned, they range from 0.98 down to 0.78.

Finally, Figures 10 and 11 present the actual and the predicted vessel traffic flow per time frame of 5 min., in the training, validation and testing sets, regarding the busiest grid cell. The traffic flow is predicted by the XgBoost model. As expected, the model performs better in the training set than in the validation set. However, it is obvious that the performance in both sets are comparable and there is no sign of overfitting. Finally, in the zoomed area the last three observations used for testing purposes are presented.

The two alternative VTFF strategies presented in the work handle the AIS data differently, which affects the forecasted vessel traffic flow. More specifically, the indirect VTFF strategy does not take into account vessel records corresponding to low speed or vessel trajectories composed of a very low

VTFF strategy	**Method**	**Time prediction horizon (min)**		
		5	10	15
Direct VTFF	XgBoost	17.72	30.41	27.43
	ARIMA	46.94	37.75	48.73
Indirect VTFF	Linear	6.35	16.76	28.71

Table 7. Prediction Results (SMAPE) in the testing set (20 busiest grid cells), G = 10km

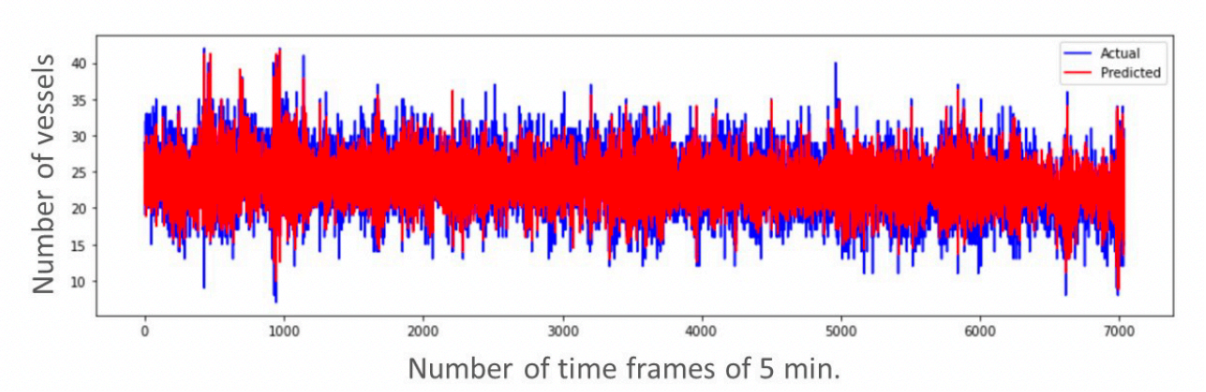


Figure 10. Actual vs. predicted vessel traffic flow produced by the VTFF with XgBoost in the busiest grid cell for the training set

(a)

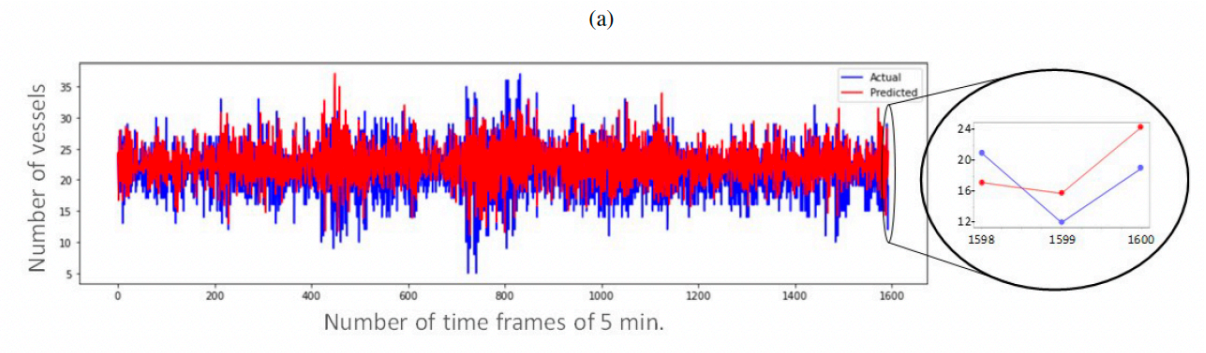


Figure 11. Actual vs. predicted vessel traffic flow produced by the VTFF with XgBoost in the busiest grid cell for validation set and testing set (zoomed graph).

Grid cell (km)	Time frame (min)	SMAPE	Jaccard
5	5	9.57	0.95
	10	26.20	0.87
	15	44.00	0.78
10	5	4.97	0.97
	10	14.23	0.93
	15	24.90	0.87
15	5	3.52	0.98
	10	10.08	0.95
	15	18.04	0.91

Table 8. Prediction results (SMAPE, Jaccard) for the indirect VTFF strategy in the testing set (All Grid cells).

number of points. As a result, the future vessel flow counts are produced mostly by vessels with significant movement, i.e., it reflects the variable traffic flow anywhere in the grid.

On the other hand, the direct VTFF strategy is able to take into account all the available vessel records, even vessel trajectories composed of only one or two records. However, due to its nature, it is able to forecast the traffic volume only within busy regions.

It should be noted that the prediction results for the case of G = 10 km grid cells produced by the indirect VTFF strategy are different between Table 7 and Table 8. In the former table the SMAPE is calculated by taking into account only the 20 busiest grid cells and in the latter table the SMAPE includes all the available grid cells. It is obvious that the prediction capability of the indirect VTFF strategy is affected by the number of non-busy regions; it predicts better when there are also non-busy regions.

4.5 Discussion

An effective VTFF method is important for improving the quality, safety, and reliability of Maritime Transport Systems. By taking advantage of AIS-derived vessel movement data, this chapter examined the VTFF problem from two different methodological perspectives: indirect forecasting, where future vessel locations are first predicted and then aggregated into traffic flow, and direct forecasting, where future traffic flow is predicted directly from historical flow sequences. The comparative study shows that both strategies can be effective for short-term maritime traffic-flow forecasting, but they exhibit different strengths, limitations, and operational implications.

A first important observation is that the distinction between direct and indirect VTFF is not merely a technical implementation choice. Rather, it corresponds to two different views of how aggregate traffic should be modeled. The indirect strategy treats future traffic flow as the result of future vessel movement. In this sense, it preserves a stronger connection between the aggregate prediction and the underlying object-level behavior. The direct strategy, by contrast, treats future traffic flow as a spatiotemporal sequence that can be learned directly from historical aggregate observations. Each view is meaningful, but each privileges a different level of abstraction.

A second observation concerns predictive behavior over short horizons. The indirect strategy performs particularly well at the earliest forecasted time frames, which suggests that, when route-prediction quality is sufficiently high, vessel-level motion carries valuable information for anticipating near-future traffic distribution. This is especially relevant in maritime settings where regional flow emerges from a relatively small number of meaningful vessel movements rather than from a very dense sensor network. However, the performance advantage of the indirect strategy weakens at longer horizons, where route-prediction errors accumulate and affect the final aggregate result. This highlights a central methodological limitation of indirect VTFF: its final performance depends on the stability of the upstream VRF stage.

A third observation concerns the role of direct aggregate forecasting. The direct strategy is computationally simpler and more operationally straightforward, because it avoids the intermediate route-prediction and aggregation stages. It can therefore be attractive in applications where the main interest lies in future regional traffic intensity rather than in the predicted paths of individual vessels. The results with XGBoost show that such direct modeling can be highly effective, especially in busy regions where the historical traffic-flow sequences contain a sufficiently strong signal. At the same time, this strength is tied to the existence of stable aggregate patterns; in sparse or structurally evolving settings, direct methods may have less informative input than indirect approaches.

A fourth observation is that spatial granularity has a strong influence on the comparative picture. The indirect strategy becomes more robust as the grid cells grow larger, because the aggregation process absorbs local route-prediction errors more effectively. At finer spatial resolution, however, small positional errors can lead to larger deviations in the final flow counts. This means that grid design is not only an experimental parameter but also a substantive modeling choice. It determines the level of abstraction at which traffic is represented and therefore influences both predictive difficulty and operational interpretability.

A fifth observation concerns evaluation itself. The chapter shows that the apparent superiority of a VTFF strategy depends partly on what exactly is being evaluated: only the busiest regions or the full spatial grid, only flow magnitude error or also vessel-set overlap, one-step prediction or several future transitions. This reinforces a broader thesis-level point already emerging from Chapter 3: mobility forecasting methods cannot be compared meaningfully without careful attention to the representation, target definition, and evaluation protocol used in each experiment.

From a systems perspective, the two strategies also imply different computational profiles. Indirect VTFF is inherently more complex because it requires a first forecasting stage at the vessel level and a second aggregation stage at the traffic-flow level. Direct VTFF is operationally leaner, but its simpler pipeline comes at the cost of a weaker explicit connection to vessel-level dynamics. Therefore, the comparison between the two strategies should not be interpreted only in terms of predictive error, but also in terms of interpretability, complexity, and deployment logic.

These findings are important for the thesis as a whole because Chapter 4 completes the transition from individual-level forecasting to aggregate-level forecasting. Chapter 3 established the benchmarking perspective for route prediction at the object level. Chapter 4 extends this logic to the aggregate level and shows that the central methodological distinction is no longer between model families alone, but between forecasting paradigms themselves. This is why the results of the chapter are significant beyond the specific dataset used in the experiments: they clarify the conceptual trade-offs that govern aggregate mobility forecasting more generally.

The chapter also directly motivates the unified forecasting direction developed in Chapter 5. If indirect VTFF preserves richer vessel-level predictive information and direct VTFF offers simpler and often more stable aggregate prediction, then a natural next question is whether these two views can be combined in a principled way. The answer to that question is not provided by choosing one paradigm over the other, but by designing a framework that attempts to exploit the complementary strengths of both. In this sense, the main contribution of Chapter 4 is not only the comparison itself, but also the methodological rationale it provides for moving toward unified forecasting.

Future work specific to this chapter includes extending the comparative study to additional environmental context, such as weather and other traffic-related variables, and examining how the two paradigms behave under longer prediction horizons, alternative grid resolutions, and larger maritime datasets. Broader thesis-level future directions are discussed in Chapter 10.

4.6 Summary

This chapter presented a systematic comparison of two major approaches to Vessel Traffic Flow Forecasting: indirect VTFF, in which future traffic flow is derived from predicted vessel trajectories, and direct VTFF, in which future traffic flow is predicted directly from historical aggregate flow sequences. By studying the two paradigms under a common spatiotemporal representation and using real AIS data, the chapter clarified their assumptions, methodological differences, and predictive behavior under realistic maritime mobility conditions.

The chapter showed that aggregate traffic forecasting can be formulated in two fundamentally different but equally meaningful ways. The indirect paradigm preserves the connection between future traffic flow and individual vessel movement, thereby capturing richer behavioral information and performing particularly well at short forecasting horizons. The direct paradigm, by contrast, models aggregate traffic flow directly as a spatiotemporal sequence and therefore offers a simpler and more operationally straightforward prediction pipeline. These differences are not superficial; they reflect two alternative abstractions of the same mobility phenomenon.

The experimental findings confirmed that neither paradigm is universally superior. The indirect strategy is especially valuable when route-level movement structure is informative and when one wishes to retain an explicit link between aggregate traffic and individual-object behavior. The direct strategy is especially attractive in busy regions where historical flow sequences provide a strong predictive signal and where simpler aggregate monitoring is of primary operational interest. The chapter also showed that the evaluation outcome depends on spatial granularity, forecast horizon, and the definition of the evaluation region, which reinforces the broader thesis argument that mobility forecasting must be studied under controlled and clearly specified experimental settings.

An important contribution of this chapter is therefore methodological rather than purely empirical. By placing direct and indirect VTFF under a common formulation, the chapter makes explicit a distinction that is often present but not sufficiently analyzed in the literature. This helps move the discussion of traffic-flow forecasting beyond isolated model comparisons and toward a clearer understanding of the trade-offs among forecasting paradigms, including their predictive, interpretive, and computational consequences.

Within the broader structure of the thesis, Chapter 4 marks the transition from object-level forecasting to aggregate-level forecasting. Chapter 3 established a controlled benchmark for Vessel Route Forecasting, whereas Chapter 4 examined how future maritime traffic can be predicted over space and time at the regional level. In doing so, it provided the aggregate forecasting foundation on which the next chapter builds.

The results of the present chapter naturally motivate the development of a unified framework. Since direct and indirect VTFF each capture useful but incomplete aspects of the forecasting problem, the next step is to examine whether vessel-level and flow-level predictive information can be combined in a common architecture. This is precisely the objective of Chapter 5, where the thesis moves from comparison to integration.

The material presented in this chapter is based on and extends the publication “Machine learning models for vessel traffic flow forecasting: An experimental comparison” [69].

Chapter 5: A Unified Approach to Vessel Traffic Flow Forecasting (UA-VTFF)

5.1 Introduction

The previous chapter studied Vessel Traffic Flow Forecasting (VTFF) through two alternative paradigms: indirect VTFF, in which future traffic flow is derived from predicted vessel trajectories, and direct VTFF, in which future flow values are predicted directly from historical aggregate flow sequences. That comparison showed that both approaches can be effective, but also that each one has important limitations. Indirect methods preserve richer vessel-level predictive information but are computationally heavier, while direct methods are operationally simpler but may lose object-level movement structure. This observation motivates the main idea of the present chapter: rather than selecting one paradigm over the other, we investigate whether their strengths can be combined within a unified forecasting framework.

In the maritime domain, safe and efficient navigation becomes increasingly important as traffic grows on sea routes and in congested waterways [64]. For this reason, accurate vessel traffic flow forecasting is valuable not only for strategic maritime planning, but also for safety-oriented applications such as traffic supervision and collision avoidance. Forecasting vessel flows can assist maritime authorities in monitoring high-risk areas and can help fleet operators reason more effectively about future regional traffic conditions.

Maritime traffic information systems already support traffic monitoring, but their forecasting capability is often still based on relatively simple prediction mechanisms or expert-driven rules [65]. Such solutions can be efficient under stable sailing conditions, but their reliability decreases when vessel behavior becomes more complex, such as during maneuvering or in dynamically evolving traffic regimes [65]. Over the last decade, machine learning methods have attracted increasing attention in maritime analytics because they can model nonlinear behavior and exploit large historical datasets more effectively [3], [4], [18], [66], [67].

As discussed in Chapter 4, many of the most promising VTFF methods rely on grid-based representations and approach the forecasting problem from either a direct or an indirect perspective [56], [69]. However, these paradigms need not be treated as mutually exclusive. The central intuition of this chapter is that direct flow-sequence modeling and indirect vessel-level aggregation may provide complementary predictive signals. A unified method that combines them may therefore improve robustness and predictive performance, especially in finer spatial resolutions and longer horizons where purely direct or purely indirect approaches may each become insufficient.

The chapter therefore introduces the Unified Approach to Vessel Traffic Flow Forecasting (UA-VTFF). The objective is not simply to build another predictive model, but to move from comparison to integration: to construct a forecasting framework that benefits both from vessel-level motion information and from region-level flow dynamics. In this sense, Chapter 5 is the natural continuation of Chapter 4, translating the earlier comparison of paradigms into a unified forecasting design.

5.2 Problem Formulation

The basic formulations of Vessel Route Forecasting (VRF) and Vessel Traffic Flow Forecasting (VTFF) were introduced in Chapters 3 and 4, respectively. Building on those definitions, the present chapter formulates a unified forecasting setting in which vessel-level route prediction and flow-level traffic prediction are combined within a single framework. The purpose of this formulation is not to redefine the two problems independently, but to specify how they interact in order to support aggregate traffic forecasting more robustly.

Consider a maritime dataset D composed of vessel trajectories, where each trajectory is represented as a sequence of timestamped locations over a maritime study area. Let the study area be partitioned into a spatial grid of cells, and let time be partitioned into consecutive temporal frames. Let Δt denote the prediction horizon and r the number of future temporal transitions up to Δt.

Using D, the unified forecasting problem addressed in this chapter is formulated as follows:

- Given:
  - a maritime dataset of vessel trajectories over a study area,
  - a spatial partition of the study area into grid cells,
  - a temporal partition into consecutive time frames,
  - a prediction horizon Δt,
  - a number of future temporal transitions r, and
  - historical traffic-flow sequences over the spatial grid together with vessel-level trajectory information

- Predict: the future traffic-flow values over the grid cells for the next r time transitions up to the prediction horizon Δt, by jointly exploiting (a) flow-level information derived from historical aggregate traffic and (b) vessel-level predictive information derived from future route forecasting.

Under this formulation, the unified approach differs from the indirect VTFF strategy of Chapter 4 because it does not rely solely on predicted vessel trajectories to produce future flow. It also differs from the direct VTFF strategy because it does not treat future traffic flow as depending only on historical aggregate flow sequences. Instead, it integrates both views in a common prediction pipeline.

More specifically, vessel-level predictions produced by the route-forecasting component are aggregated into future traffic-related signals over the grid, while the direct VTFF component models historical traffic dynamics over the same spatial partition. These sources of information are then combined at the feature level in order to improve the prediction of future traffic-flow values, especially in settings where either direct-only or indirect-only forecasting may be insufficient on its own.

5.3 Methodology

In this section, our approach is presented including the enhanced VRF model and the UA-VTFF methodology. An overview of the proposed methodology is illustrated in Figure 12. More specifically, first, historical AIS data pass through processing operations. Then the VRF method is applied. The processed historical data along with the produced points of the VRF are arranged into a spatiotemporal grid. Subsequently, the resulting traffic flows pass through a feature vector analysis, which are then fed to an ML method for VTFF prediction purposes.

5.3.1 The enhanced VRF model

In this work, we consider the VRF problem as a direct trajectory forecasting problem. More specifically, in order to train the model, the output data are interpolated on the trajectory's r transitions. Also, the NN consists of r * 2 outputs, where for one predicted point, there are necessary two neurons for the two coordinates

for space and, as a result, each pair of output neurons gives one forecast for each trajectory's transition. Thus, after the training procedure, we need to execute the NN model only one time in order to produce the desired predictions.

As far as the input information is concerned, three features are given including information regarding the time and the two space coordinates, actually the differences in time and space between consecutive timestamped positions. More specifically, for each timestep $k = 1, \ldots, n_{j-1}$, where n_j is the length of the j-th trajectory, the NN is fed with the input vector:

$$\Delta \boldsymbol{u}_j^s(k) = \left[\Delta x_j^s(k), \Delta y_j^s(k), \Delta t_j^s(k)\right]. \quad (9)$$

Regarding the NN architecture, as depicted in Figure 12, it consists of an input layer of three neurons, a Long Short-Term Memory (LSTM) [35] hidden layer, a fully connected hidden layer and an output layer of r * 2 neurons. Also, the model parameters can be updated through the Backward Propagation Through Time (BPTT) algorithm [37] and the synaptic weights can be optimized according to the Adam approach [38] in the training set. Overall, the NN training phase is followed by a validation phase and, eventually, model selection; the use of a validation set results in measuring the model's generalization ability.

It should be noted that the VRF problem in [69] was considered as a multipoint location forecasting task and the model presented in [57] was employed for predicting the vessel's future locations. This model (after following the training procedure) needs to be executed r times in order to provide predictions for the r transitions forming the future trajectory. On the other hand, our proposed method needs to be executed only one time to produce the desired trajectory and thus the time prediction duration significantly decreases, as it will be shown in the experimental study.

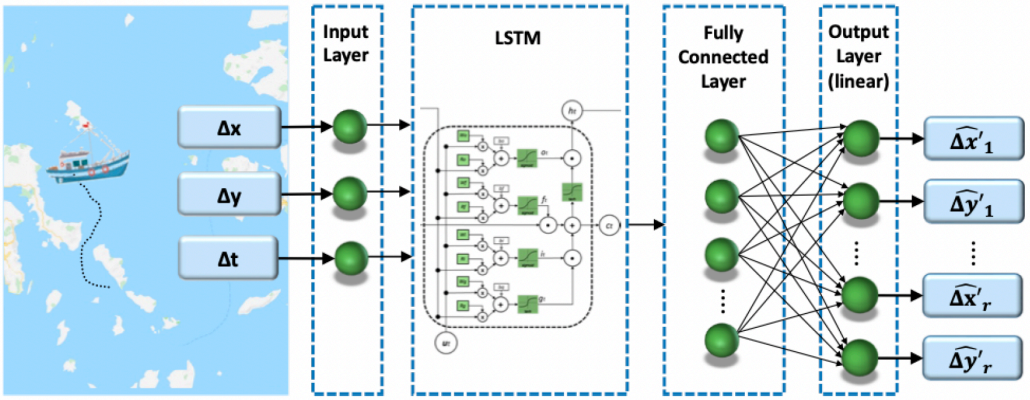


Figure 12. Proposed VRF-LSTM network architecture, with one input layer, one LSTM layer, one fully-connected layer, and one output layer.

5.3.2 The Unified Approach for VTFF (UA-VTFF)

The intuition behind UA-VTFF is to combine the complementary strengths of the two forecasting paradigms examined in Chapter 4. The indirect VTFF component contributes vessel-level predictive signal by forecasting future individual movement and then aggregating these predictions spatially, while the direct VTFF component captures temporal regularities already visible at the flow level. UA-VTFF integrates these two views into a single forecasting framework, with the aim of improving robustness in sparse regions and enhancing predictive accuracy under dynamically changing traffic conditions.

UA-VTFF method starts by giving the available vessels' historical trajectories into the proposed VRF algorithm (described in the previous section) in order to predict the future trajectories. Subsequently, the vessels' historical and future trajectories are assigned into the spatiotemporal grid and the calculation of the

amount of vessels that are placed in every grid cell is performed. The resulting amounts indicate the volume of the historical vessels n and the predicted vessels η (produced by the proposed VRF model), which represent the traffic sequence in a specific cell-region and time frame. Then, these traffic sequences for every grid cell b are enhanced with additional features, including information regarding timestamp and the volume of vessels in surrounding cells. Finally, the produced sequences are fed to an ML algorithm for predicting the future traffic flow in the grids, i.e. the vessels' amount in a particular cell in future time frame Δt. Figure 13 presents the above pipeline.

It should be noted that for training purposes the data in the output are also processed as the data in the input. More specifically, in order to predict the vessels' amount in the b-th grid cell in the future time frame $t + 1$, i.e., to produce the model output $n\hat{}b\ t{+}1$, the corresponding model input $Nb\ t{+}1$ for the b-th cell within the grid that comprises l total cells is composed of the following features:

- $n^{b}_{\{t-l\}}, \dots, n^{b}_{\{t-1\}}, n^{b}_{t}$: the vessels' number n in time frame t in the b-th cell grid
- $n^{\{b'\}}_{\{t-l\}}, \dots, n^{\{b'\}}_{\{t-1\}}, n^{\{b'\}}_{t}$: the vessels' number n in time frame t in the neighbor-surrounding cell grids b' of the b-th cell grid
- $n^{\{b''\}}_{\{t-l\}}, \dots, n^{\{b''\}}_{\{t-1\}}, n^{\{b''\}}_{t}$: the vessels' number n in time frame t in the neighbor-surrounding cell grids with high traffic b'' of the b-th cell grid
- $\eta^{b}_{\{t+1\}}$: the vessels' number produced by the VRF-LSTM model in the future time frame $t{+}1$ in the b-th cell grid
- $\eta^{\{b'\}}_{\{t+1\}}$: the vessels' number produced by the VRF-LSTM model in the future time frame $t{+}1$ in the neighbor-surrounding cell grids b' of the b-th cell grid
- $\eta^{\{b''\}}_{\{t+1\}}$: the vessels' number produced by the VRF-LSTM model in the future time frame $t{+}1$ in the neighbor-surrounding cell grids with high traffic b'' of the b-th cell grid
- $md^{b}_{\{t-l\}}, \dots, md^{b}_{\{t-1\}}, md^{b}_{t}$: the day of the month md in each time frame t in the b-th cell grid
- $dw^{b}_{\{t-l\}}, \dots, dw^{b}_{\{t-1\}}, dw^{b}_{t}$: the day of week dw in each time frame t in the b-th cell grid
- $td^{b}_{\{t-l\}}, \dots, td^{b}_{\{t-1\}}, td^{b}_{t}$: the time of day td in each time frame t in the b-th cell grid

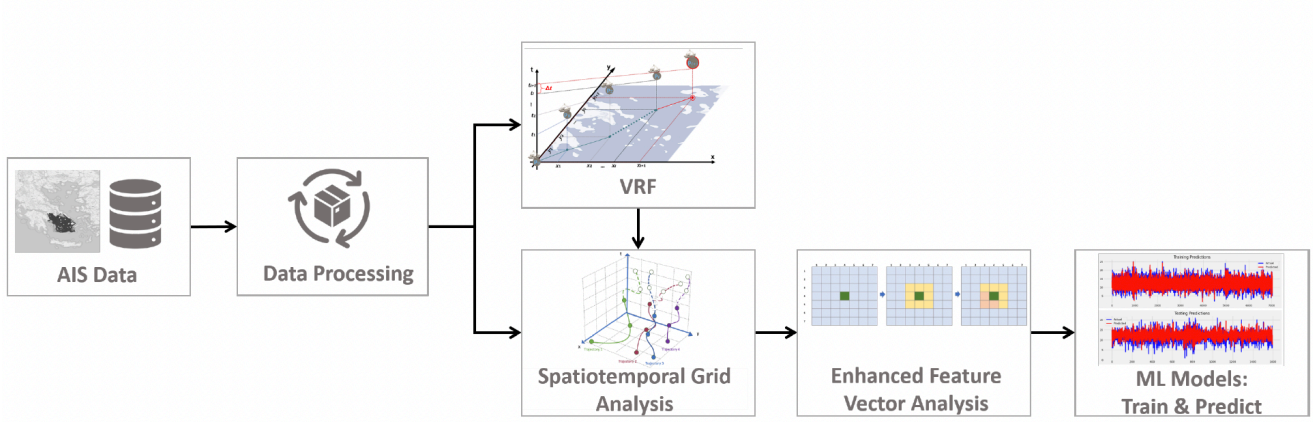


Figure 13. Overview of the UA-VTFF methodology

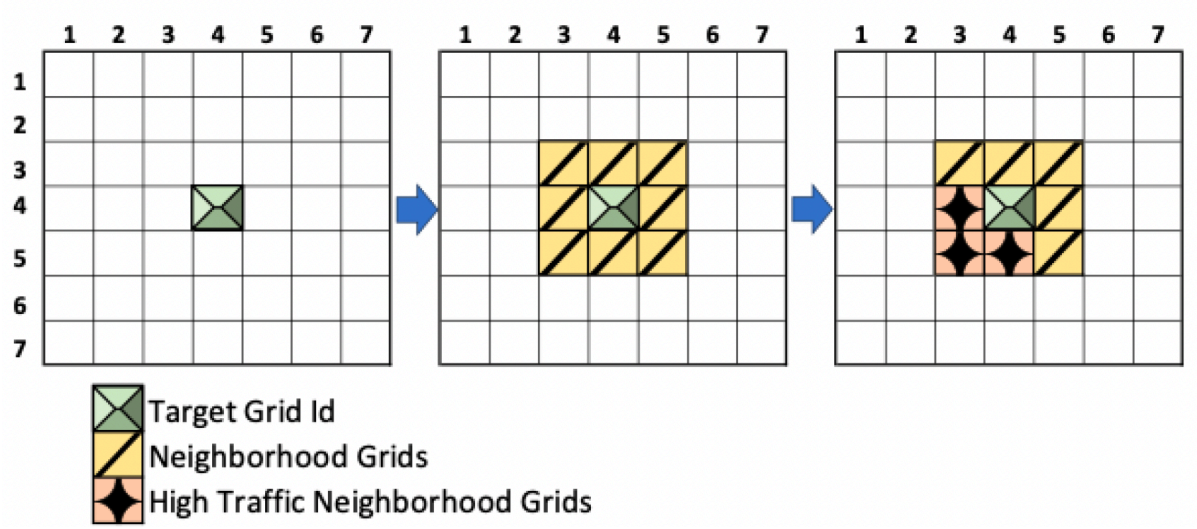


Figure 14. Example of 7x7 neighbor-surrounding traffic cells of a spatial grid.

Regarding the calculation of the neighbour-surrounding grid cells and those with high traffic, the neighbour cells correspond to those cells that surround the center-targeted. The surrounding grid cells with the highest traffic are those with the highest correlation with the center-targeted and are defined based on statistical analysis and taking into account that these cells should correspond to regions areas associated with a high risk of accidents [51]. To clarify the above discussion, Figure 14 illustrates a grid with nearby cells only in space of 7x7. The green cell illustrates the central-targeted cell for which a prediction will take place. The yellow cells correspond to the neighbourhood cells, while the pink cells present the neighbourhood cells with high traffic and high correlation with the center-targeted.

Several ML techniques were used in this study to tackle the VTFF problem. Particularly, we employ XgBoost and ARIMA models, which were trained according to [69]. Also, we employ Facebook Prophet [74], which is a recently presented forecasting technique inspired by the nature of time series predicting at Facebook.

Furthermore, we use static and dynamic NNs. Particularly, for the static NNs, the Multi-Layer Perceptron (MLP) architecture was employed composed of two hidden neurons and trained with the backpropagation algorithm [33]. Regarding the dynamic NNs, LSTM networks were used composed of an input layer, an LSTM hidden layer, a fully-connected hidden layer and an output layer of two neurons. The LSTM-based architecture was trained following the same procedure as it was described in Section 4.1.

As far as the model parameterization is concerned, various characteristics of every model type were taken into account and adjusted with intermediate observation tests before the final evaluation. In particular, the XgBoost models were optimized regarding the learning rate, the minimum leaf size for pruning, the number of features on a node, and the number of regression trees. Also, the ARIMA model was optimised regarding the lag order, the degree of differencing, and the order of the moving average by evaluating the Partial Autocorrelation plot, the Augmented Dickey Fuller test and the Autocorrelation plots, respectively. Furthermore, the MLP and LSTM NNs were optimised regarding the hidden layers' size, while an early stopping procedure [46] is employed to prevent the networks from overfitting by using a validation set. As far as the Facebook Prophet parameterization is concerned, two parameters were tuned that express Seasonality variance and Trend variance: Seasonality Prior Scale influences the seasonality component of the time series and the values tested ranged from 0.01 (small influence) to 10 (very high influence); Changepoint Prior Scale

influences the variance of the trend component and the values tested ranged from 0.05 (underfitting variance) to 0.5 (overfitting variance).

5.4 Experimental Setup

All the methods were implemented using Python and the experiments were conducted in a workstation composed of 64 GB RAM, Intel Core i9-9900KX CPU and GeForce RTX2080Ti GPU.

The proposed method was evaluated on real-world AIS data provided by MarineTraffic.com. More specifically, 1,757,440 AIS records received from 2344 different vessels of various types sailing during November, 2018, in the Aegean Sea rectangle bounded by longitude [23...26] and latitude in [36...38].

As far as the parameters of the VTFF problem formulation are concerned, we used a spatial grid of G = 2km, a prediction horizon of Δt = 30min and a number of temporal transitions of r = 6. The produced traffic flow is as follows: A number of 14,340 cells was created, with the traffic flow of min, max, median and mean values equal to 0, 103617, 25, and 120 vessels, respectively. Of the whole grid cells, only 768 cells presented traffic flow of more than 300 vessels across the entire period, while 4,000 cells included less than 10 vessels in the whole period. The top 10 grid cells have more than 5 vessels on average every 5 minutes and include more than 300,000 vessels in the entire period.

It should be noted that we resulted in the abovementioned parameters (G = 2km, Δt = 30min and r = 6) after experimenting with different values of spatial grid's resolution (of G equal to 2km, 5 km and 10 km), number of temporal transitions (of r equal to 3 and 6) and prediction horizon (of Δt equal to 15 min. and 30 min.; based on the tested values of r the overall corresponding time frames were of 5 min. and 10 min.). We observed that the UA-VTFF method is affected by these parameters. More specifically, the model predicts better when the spatial resolution (based on G) is higher and the time frames (based on r in combination with Δt) are bigger, while the impact of the prediction horizon is limited.

Regarding the evaluation of the implemented algorithms, it was performed in the busy grid cells [85], which correspond to regions with regular navigation and heavy traffic, with a high risk of accidents [51]. Areas with low vessel density denote a low level of traffic flow complexity, which implies regular and predictable vessel travel time sequences [86]. This is also confirmed by the experimental study in [69], where the ML model predicts better when there are also non-busy regions; the same ML model was tested on the whole available cells (there were cells with zero traffic) and only on the busy ones.

For all the algorithms, except the indirect VTFF [69] , the corresponding traffic flow sequences of the busy grid cells are arranged into input and output data, where for each grid cell, the initial 75% of the traffic flow sequence is used for the training purpose, whereas the remaining 25% of the traffic flow sequence, except for the last six observations (corresponding to 30 min.), is organized in the testing set. Regarding the indirect VTFF, following [69] the trajectories derived from the busy grid cells are then arranged into training, validation, and testing sets of a 50%–25%–25% percent ratio, respectively. Experimental results are evaluated using the Symmetric Mean Absolute Percentage Error (SMAPE) [69].

5.5 Results

Table 9 presents results for the UA-VTFF approach using different ML techniques. More specifically, LSTM, MLP, XgBoost, ARIMA and Prophet are employed. Also, Table 8 presents results for the UA-VTFF approach, the indirect and direct VTFF strategies presented in [69].

The results presented in Table 9 confirm that the LSTM-based UA-VTFF method can accurately capture the vessel traffic flow in short-term to assist in effective collision avoidance. This is due to the fact that LSTM have emerged as an effective technique for several difficult learning problems [3] [5] and have demonstrated

significant performance on predicting sequential problems [87]. Also, our proposed UA-VTFF approach (using LSTM or XgBoost) outperforms the indirect and direct VTFF strategies as summarized in Table 10.

In order to investigate further the performance of the proposed UA-VTFF, we also present results in the two highest traffic grid cells. Particularly, Figures 15a and 15b depict the prediction performance of the direct VTFF strategy [69] and the XgBoost-based UA-VTFF algorithm, respectively in the grid cell with id #19837. Also, Figures 16a and 16b depict the prediction performance of the direct VTFF strategy [69] and the XgBoost-based UA-VTFF algorithm, respectively in the grid cell with id #19653. More specifically, the figures illustrate the original and predicted traffic flow in the training and testing sets. The figure for the training graphic corresponds to the number of vessels for the first 75% of the 5 min time frames(first 7000 time frames), while the testing graphic corresponds to the number of vessels for the last 25% of the 5 minutes time frames (last 1600 time frames).

Method	**Time prediction horizon (min)**					
	5	10	15	20	25	30
LSTM	12	16	27	26	27	26
MLP	16	21	32	34	37	36
XgBoost	13	19	29	29	29	27
ARIMA	20	24	35	38	47	40
Prophet	21	24	33	33	38	36

Table 9. Prediction results (SMAPE) for different alternatives of the UA-VTFF method in the testing set (G=2km)

Approach	**Train set**	**Test set**
UA-VTFF (LSTM	18	21
UA-VTFF (XgBoost)	19	22
direct VTFF	23	29
indirect VTFF	25	28

Table 10. Prediction results (SMAPE) for each approach (G = 2km)

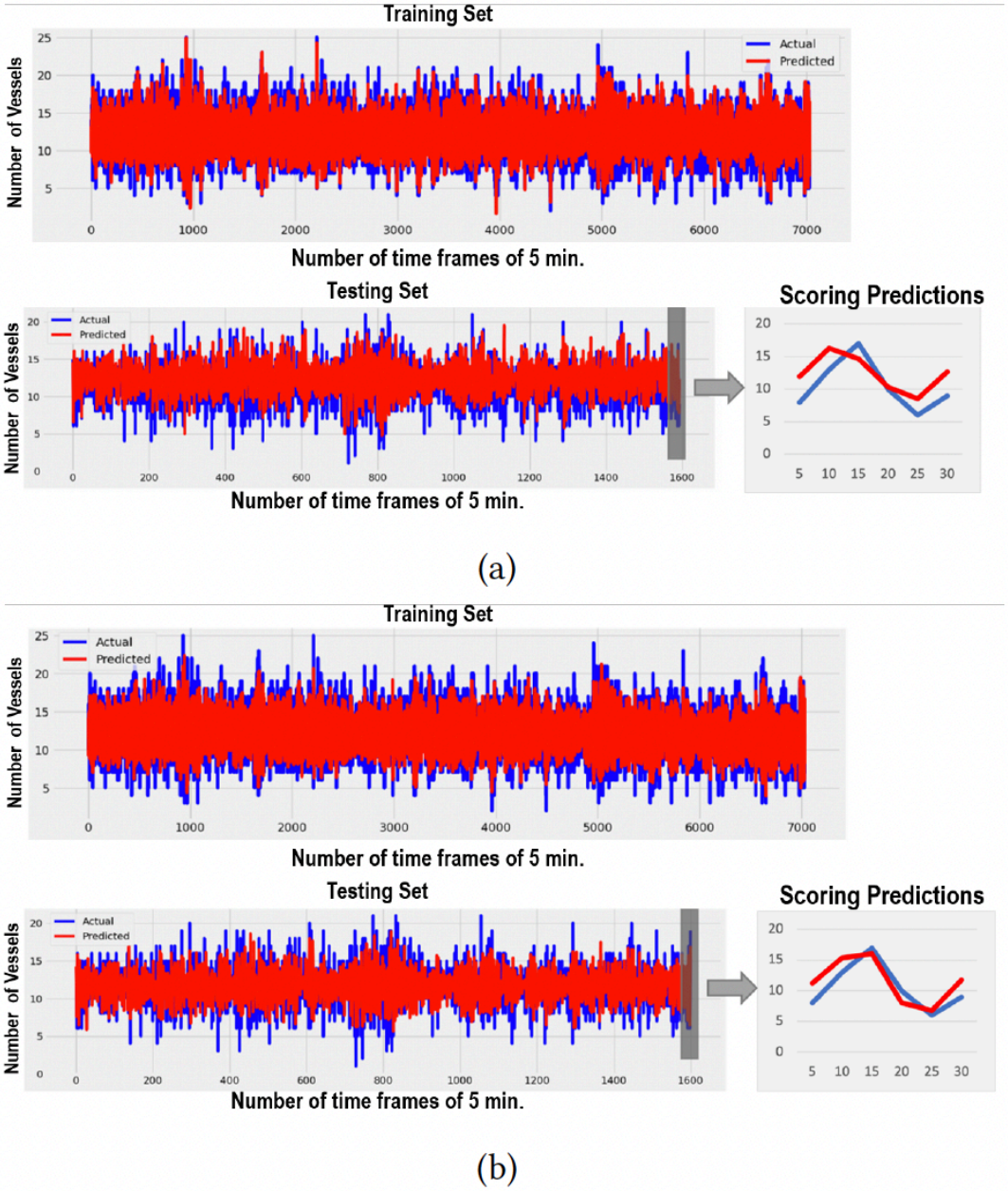


Figure 15. Prediction performance in training and testing sets for grid cell id 19837 of the: (a) XgBoost-based direct VTFF, and (b) XgBoost-based UA-VTFF

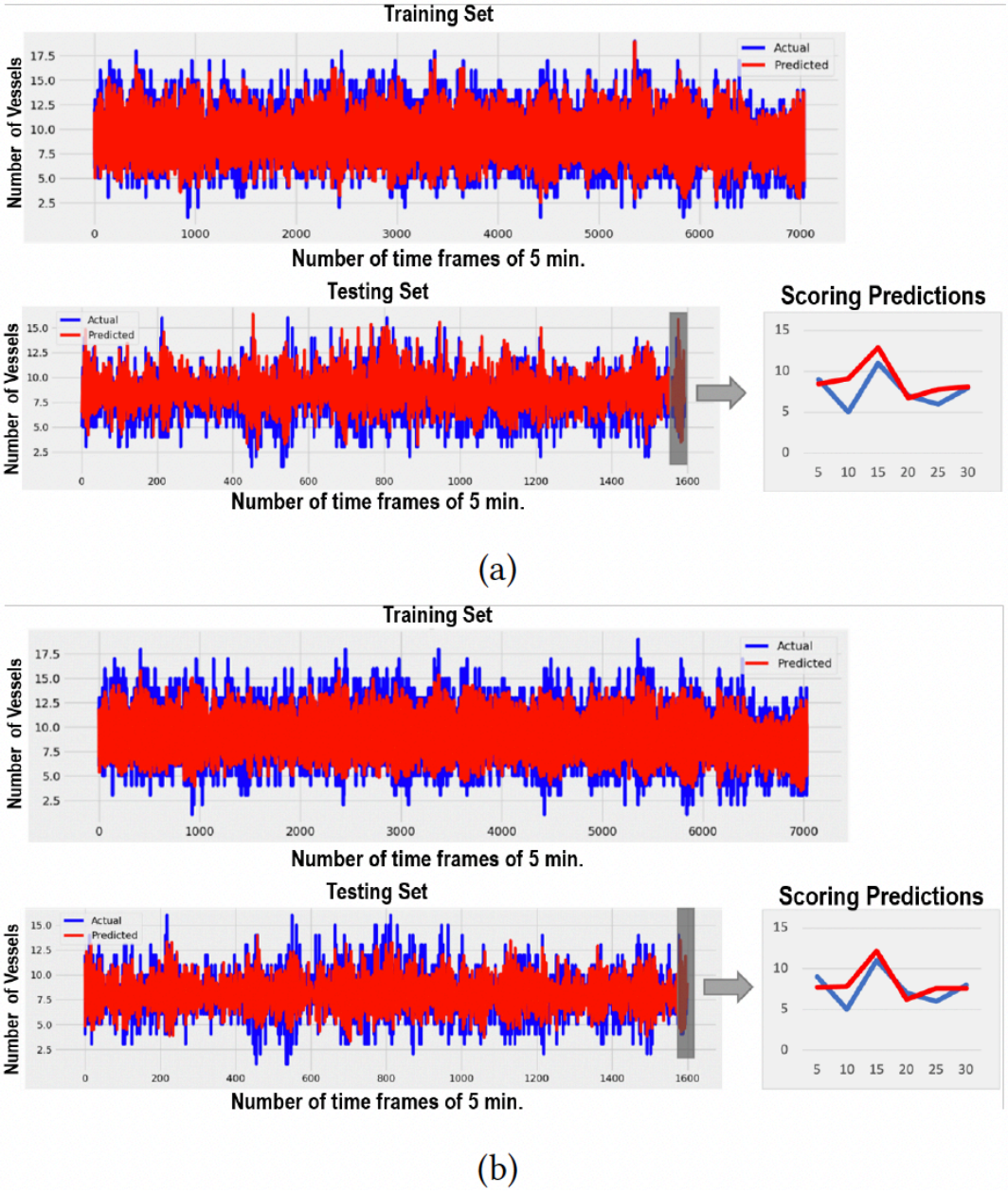


Figure 16. Prediction performance in training and testing sets for grid cell id 19653 of the: (a) XgBoost-based direct VTFF, and (b) XgBoost-based UA-VTFF

Finally, we have included the performance of each method on the last six observations of the dataset corresponding to the last 30 minutes. Based on Figures 15b, 15a, 16b and 16a we can observe that UA-VTFF has less overfitting in the training data compared to the direct VTFF [69], since in the latter method the training graphic seems to have a much better performance than the testing one. Finally, the XgBoost-based UA-VTFF method performs better in the scoring dataset than the direct VTFF, which also uses XgBoost models. Finally, we should mention that the proposed VRF method manages to predict a vessel trajectory composed of 6 transitions with a latency of about 0.5secs on average. On the other hand, the VRF method used in the indirect VTFF [11] predicts 6 transitions with a latency of about 2.5 secs on average.

5.6 Discussion

The results of this chapter show that the unified forecasting direction is methodologically justified and practically promising. Chapter 4 demonstrated that direct and indirect VTFF provide different but complementary views of aggregate traffic forecasting. The present chapter moves one step further by showing that these two views need not remain separate. Instead, they can be integrated in a common framework that exploits both historical flow structure and vessel-level predictive information.

A first important observation is that the unified approach is not simply a hybrid in the superficial sense of combining multiple features. Its value lies in the fact that it addresses a real limitation identified in the previous chapter. Direct VTFF is operationally simple and often effective in busy regions, but it can be less informative when aggregate historical patterns alone are insufficient. Indirect VTFF preserves richer movement-level information, but it is vulnerable to the accumulation of route-prediction errors and can be computationally heavier. UA-VTFF is motivated precisely by the need to balance these two weaknesses while retaining their strengths.

A second observation concerns the role of vessel-level predictive information inside an aggregate forecasting task. The chapter shows that route-level predictions can enrich flow-level modeling when they are incorporated in a structured way rather than treated as a separate end product. This is particularly relevant in mobility settings where future aggregate traffic does not emerge only from historical regional counts, but also from the evolving movement of individual objects. In this sense, the unified framework supports a more informative representation of future traffic than direct aggregate modeling alone.

A third observation is that the unified framework should also be interpreted in systems terms. The enhanced VRF model introduced in this chapter is important not only because it improves the efficiency of route prediction, but also because it makes the unified architecture more feasible operationally. If vessel-level forecasts are to be used as internal components of aggregate prediction, then prediction latency and execution cost become critical design issues. Therefore, the contribution of the chapter lies not only in combining two paradigms conceptually, but also in making that combination computationally more practical.

A fourth observation is that the unified perspective changes the way aggregate forecasting should be evaluated. Rather than asking whether direct or indirect VTFF is universally superior, the chapter suggests that the more meaningful question is under what conditions each source of predictive information contributes useful signal. This is a more thesis-level view of the problem, because it moves beyond isolated model comparisons and toward a clearer understanding of how forecasting architectures should be designed under realistic mobility-data conditions.

The chapter also has limitations that should be acknowledged. Although the results are promising, the evaluation remains centered on a specific maritime AIS setting and on a particular unified design. Different spatial resolutions, longer prediction horizons, or alternative methods for combining vessel-level and flow-level information may lead to different trade-offs. In addition, the computational demands of unified forecasting remain nontrivial even with the enhanced VRF component. These limitations motivate the next chapter.

Within the broader structure of the thesis, Chapter 5 is important because it marks the transition from comparison to integration. Chapter 4 established that direct and indirect VTFF embody different forecasting abstractions with different strengths. Chapter 5 shows that these abstractions can be combined inside a single framework. This progression is central to the dissertation, because it demonstrates how the thesis moves from benchmarking individual paradigms to designing more general forecasting architectures.

Finally, the chapter provides the conceptual basis for Chapter 6. If unified forecasting is useful, the next research question is whether it can remain effective when data volume increases, the experimental setting becomes larger, and computational feasibility becomes a central concern. This is precisely the direction developed in the distributed and Transformer-based framework of the next chapter.

5.7 Summary

This chapter introduced the Unified Approach for Vessel Traffic Flow Forecasting (UA-VTFF), a framework that combines route-level and flow-level predictive information in a common aggregate forecasting pipeline. Building on the comparison of direct and indirect VTFF in Chapter 4, the chapter addressed the question of whether the complementary strengths of the two paradigms can be exploited jointly rather than treated separately.

To support this unified design, the chapter first introduced an enhanced VRF component that produces future trajectory predictions more efficiently than the earlier multipoint formulation. This component was not treated as an isolated goal, but as an enabling mechanism for the unified framework. Its role is to provide vessel-level predictive information in a form that can be incorporated effectively into aggregate traffic-flow forecasting.

The results showed that the unified perspective is both methodologically meaningful and empirically useful. By enriching direct flow-sequence modeling with vessel-level predictive signal, UA-VTFF provides a more robust view of future aggregate traffic than either direct-only or indirect-only forecasting alone. In this sense, the chapter extends the thesis from the comparative analysis of forecasting paradigms to the design of integrated forecasting architectures.

An important contribution of the chapter is that it clarifies the relationship between route forecasting and traffic-flow forecasting within a single mobility-analytics pipeline. Rather than treating these as separate research problems, UA-VTFF shows that they can be connected in a principled manner under a shared spatiotemporal representation. This is a key step in the broader research progression of the thesis.

At the same time, the chapter also makes clear that unified forecasting introduces new computational demands. Combining vessel-level prediction with aggregate flow modeling creates a richer forecasting framework, but also a more complex one. This naturally motivates the next step of the thesis, namely the study of distributed and scalable unified forecasting under larger AIS datasets and stronger computational constraints.

For this reason, Chapter 6 extends the present framework toward distributed processing and Transformer-based modeling. In this sense, Chapter 5 serves as the conceptual bridge between the paradigm comparison of Chapter 4 and the scalable unified forecasting architecture of Chapter 6.

The material presented in this chapter is based on and extends the publication “Towards a unified vessel traffic flow forecasting framework” [93].

Chapter 6: Distributed Unified Vessel Traffic Flow Forecasting (dUA-VTFF)

6.1 Introduction

The previous chapter introduced UA-VTFF as a unified framework that combines direct and indirect vessel traffic flow forecasting within a single predictive design. While that framework improved predictive performance, it also highlighted an important practical limitation: as data volumes, feature complexity, and forecasting components increase, unified forecasting pipelines must also remain computationally feasible. This issue is especially important in big mobility data settings, where AIS streams contain large numbers of records, continuous updates, and substantial spatiotemporal heterogeneity.

We therefore move in this chapter from a unified forecasting framework to a distributed unified forecasting framework. The goal is not only to improve predictive accuracy, but also to support large-scale execution under realistic AIS workloads. In this sense, Chapter 6 is the natural continuation of Chapters 4 and 5: Chapter 4 compared forecasting paradigms, Chapter 5 unified them, and Chapter 6 addresses their scalability through distributed processing and a more expressive forecasting architecture.

As maritime data availability continues to grow due to the widespread use of the Automatic Identification System (AIS), forecasting vessel traffic flows has become both more feasible and more demanding. Recent developments in sequence modeling suggest that attention-based models such as Transformers [95] can capture long-range dependencies effectively and often outperform recurrent architectures in structured time-series problems [94], [98], [99]. Transformer-based designs have already shown promise in vessel trajectory forecasting [90], [97] and in traffic forecasting more broadly [88], [99], [104]–[106].

At the same time, large-scale forecasting requires more than model expressiveness alone. The overall forecasting pipeline must also support efficient preprocessing, flow construction, distributed feature handling, and scalable execution. Apache Spark [110] is therefore particularly relevant in this context, as it is widely used for large-scale data processing and has already been applied in maritime analytics [3], [14] as well as in broader traffic forecasting settings [101].

To address these requirements, this chapter introduces the distributed Unified Approach for VTFF (dUA-VTFF). The method extends UA-VTFF in two directions. First, it adopts a Transformer-based architecture for traffic-flow forecasting in order to better capture long-range temporal structure. Second, it embeds the forecasting workflow within a distributed processing environment based on Apache Spark, enabling more scalable execution on large AIS datasets. Thus, the chapter treats predictive performance and computational feasibility jointly rather than as separate concerns.

Overall, the main goal of this chapter is to design a distributed pipeline for vessel traffic flow forecasting that improves both predictive quality and execution efficiency under realistic AIS data conditions. In this sense, dUA-VTFF is positioned as the big-data extension of the unified forecasting framework introduced in the previous chapter.

6.2 Problem Formulation

The basic formulations of Vessel Route Forecasting (VRF), Vessel Traffic Flow Forecasting (VTFF), and the Unified Approach for VTFF (UA-VTFF) were introduced in Chapters 3, 4, and 5, respectively. Building on those definitions, the present chapter formulates a distributed unified forecasting setting in which vessel-level route prediction and flow-level traffic prediction are integrated within a common framework designed for larger data volumes and stronger computational requirements.

Consider a maritime dataset D composed of vessel trajectories over a study area, where each trajectory is represented as a sequence of timestamped locations. Let the study area be partitioned into a spatial grid of cells, and let time be partitioned into consecutive temporal frames. Let Δt denote the prediction horizon and r the number of future temporal transitions up to Δt.

Using D, the distributed unified forecasting problem addressed in this chapter is formulated as follows:

- Given:
  - a maritime dataset of vessel trajectories over a study area,
  - a spatial partition of the study area into grid cells,
  - a temporal partition into consecutive time frames,
  - a prediction horizon Δt,
  - a number of future temporal transitions r,
  - historical traffic-flow sequences over the spatial grid,
  - vessel-level trajectory information that can be used for future route prediction, and
  - a distributed processing environment capable of supporting large-scale preprocessing, learning, and inference,

- Predict: the future traffic-flow values over the grid cells for the next r time transitions up to the prediction horizon Δt, by jointly exploiting route-level and flow-level predictive information under a scalable and computationally feasible forecasting architecture.

Under this formulation, the present chapter extends the unified forecasting logic of Chapter 5 in two directions. First, it replaces the earlier route-forecasting component with a VRF-LSTM design that is better aligned with large-scale integration inside the forecasting pipeline. Second, it introduces a distributed and Transformer-based architecture in order to address the computational demands that arise when unified forecasting is applied to larger AIS datasets and more demanding experimental settings.

The objective is therefore not only to improve aggregate traffic-flow prediction, but also to ensure that such prediction remains feasible when data scale, spatial resolution, and model complexity increase. In this sense, the chapter treats scalability as a core dimension of the forecasting problem rather than as a secondary implementation detail.

6.3 Methodology

In this section, our approach is presented including the VRF-LSTM method and the dUA-VTFF methodology. An overview of the proposed methodology is illustrated in Figure 17. More specifically, first, historical AIS data pass through typical AIS data preprocessing. Then the VRF-LSTM method is applied. The processed historical data along with the produced points of the VRF are converted into Resilient Distributed Datasets (RDDs) and are arranged into a spatiotemporal grid, where each RDD representing traffic flows. Subsequently, the resulting traffic flows pass through a feature vector analysis, to a big data distributed processing framework, where each node in the distributed system is allocated a specific subset of grid cells. Finally, Transformer models are being employed for VTFF prediction.

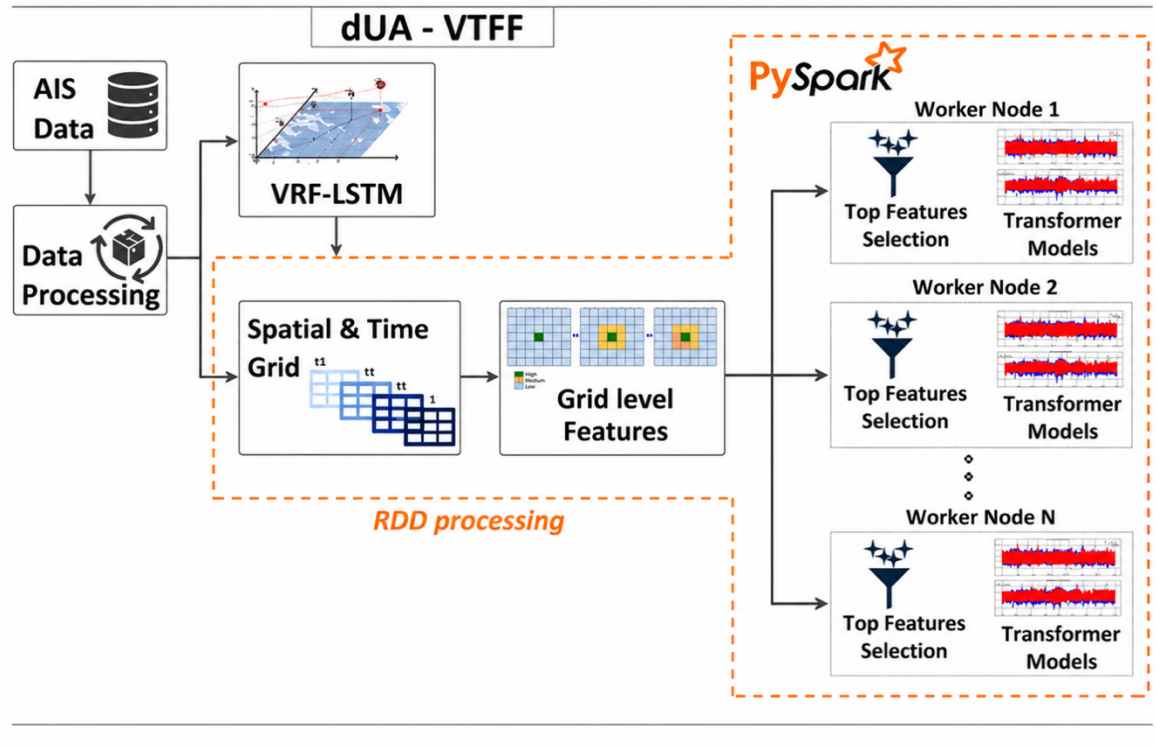


Figure 17. Overview of the dUA-VTFF methodology.

6.3.1 VRF-LSTM method

In this chapter, the VRF-LSTM component is not introduced as a new contribution in its own right, but is reused as a building block within the broader dUA-VTFF pipeline. Since the distributed unified framework retains an indirect forecasting branch based on vessel-level route prediction, a route-forecasting model is required as one of its components. For this purpose, we employ the VRF-LSTM method introduced in [93], adopting the same preprocessing, network structure, optimization, validation, and model-selection principles.

The choice of this component is motivated by earlier findings. The underlying VRF-LSTM family was originally introduced in [107] for vessel location forecasting and later adapted in [57] for multi-point trajectory forecasting under realistic AIS conditions. Compared with earlier variants, the formulation used in [93] is particularly suitable here because it can predict the required future trajectory points in a single execution of the trained model, which is important in a larger unified and distributed forecasting framework.

The overall process of producing a trained VRF-LSTM model, involves the training of the NN-based model, followed by the phases of validation and model selection [94,57,107]. First, the available data undergo two crucial preprocessing steps essential for the VRF-LSTM model, as discussed in [57]: i) simplifying stationary data (excluding records with speeds less than one knot) and eliminating insignificant trajectories (those with fewer than ten points), and ii) segmenting trajectories into sub-trajectories when the time interval between two consecutive points of the same trajectory exceeds a specific threshold (in this work set at 30 min.). Subsequently the differences in time and space between consecutive timestamped positions are computed.

The VRF-LSTM model is based on an NN model composed of an input layer of three neurons, an LSTM hidden layer, a fully-connected hidden layer and an output layer of $r * 2$ neurons. The NN predicts the future trajectories composed of 2D points of r transitions, by using as input timestamped locations. Consequently, every pair of output neurons yields a forecast for each transition in the trajectory. To facilitate model training,

the output data are interpolated along the trajectory's r transitions. Following the training process, a single execution of the NN model is adequate to generate the desired predictions.

Regarding the input information, three features are provided, encompassing details about time and two spatial coordinates. These features specifically denote the temporal and spatial differences between successive time-stamped positions. More specifically, for the s-th vessel, each timestep k = 1, . . . , n_{j-1}, where n_j represents the length of the j-th trajectory, the NN is supplied with the following input vector:

$$\Delta u_j^s(k) = \left[\Delta x_j^s(k), \Delta y_j^s(k), \Delta t_j^s(k)\right] = \left[x_j^s(k) - x_j^s(k-1), y_j^s(k) - y_j^s(k), t_j^s(k) - t_j^s(k)\right]. \quad (10)$$

The desired output of the model, which corresponds to the s-th vessel's future trajectory consisting of a total of r transitions interpolated with fixed sampling rate equal to Δt/r is the following:

$$[\Delta o'^s_j(k+1)), \Delta o_j'^s(k+2), \ldots, \Delta o_j'^s(k+r)] = [(\Delta x_j'^s(k+1), \Delta y_j'^s(k+1)), \ldots,$$
$$(\Delta x_j'^s(k+r), \Delta y_j'^s(k+r))]. \quad (11)$$

where Δx′ and Δy′ are the intervals of the interpolated positions of easting and northing, respectively. Once the prediction procedure has been completed, the NN model has predicted for the s-th vessel the easting and northing intervals Δ^x′ and Δ^y′, respectively, of the r consecutive points with fixed sampling rate equal to Δt/r, which is defined as:

$$[\Delta \widehat{o'}^s_j(k+1)), \Delta \hat{o}_j'^s(k+2), \ldots, \Delta \hat{o}_j'^s(k+r)] = [(\Delta \hat{x}_j'^s(k+1), \Delta \hat{y}_j'^s(k+1)), \ldots, (\Delta \hat{x}_j'^s(k+r), \Delta \hat{y}_j'^s(k+r))]. \quad (12)$$

Subsequently, the model output is transformed from differences to actual values. The training goal is the same as in [94,57,107], i.e. to minimize the loss function, which is defined as the Euclidean distance between the desired positions and the predicted positions. Details for the Backward Propagation Through Time (BPTT) algorithm, which was employed for updating the network parameters, can be found in [37]. Additionally, details for the Adam approach, which was employed for optimizing the synaptic weights on the training set, can be found in [38].

The training phase of the model is followed by the phases of validation and model selection [94,57,107]. The inclusion of a validation set is crucial for assessing the generalization ability of the model. In this context, the NN that exhibits superior performance over a specific number of epochs on the validation set is ultimately chosen. In simpler terms, the training concludes when the error fails to decrease for a specified number of epochs on the validation set.

6.3.2 The distributed Unified Approach for VTFF (dUA-VTFF) based on Transformer Networks

In this section the centralised and distributed versions of the Unified Approach for VTFF methods are presented.

6.3.2.1 The centralised Unified Approach for VTFF (UA-VTFF) method

Due to transceiver errors, AIS records may exhibit inaccuracies [109]. Consequently, the centralised UA-VTFF method [94] starts by applying a standard data cleansing preprocessing procedure to the available AIS data. This involves deduplication of records and the removal of noisy entries (particularly those associated with speeds exceeding 50 knots).

Subsequently, the centralised UA-VTFF method [109] proceeds by giving the vessels' historical trajectories into the proposed VRF-LSTM method (described in the previous section) in order to predict the future trajectories. Subsequently, the vessels' historical and future trajectories are assigned into the spatiotemporal grid, namely the calculation of the number of vessels that are placed in every grid cell is performed. The resulting numbers indicate the volume of the historical vessels n and the predicted vessels η (produced by the proposed VRF-LSTM model), which represent the traffic sequence in a specific cell (region) and time frame. Then, these traffic sequences for every grid cell b are enhanced with additional features, such as the volume of vessels in surrounding cells. Finally, the produced sequences are fed to an ML algorithm for predicting the future traffic flow in the grids, i.e. the vessels' number in a particular cell in future time frame Δt.

The UA-VTFF method can predict the number of vessels in the future time frame t + 1 in the b-th grid cell composed of l total cells. For this, a model should be constructed with output $\hat{n}_{b\,t+1}$ and the corresponding input $N_{b\,t+1}$ composed of the features described in [93].

Regarding the calculation of the neighbour-surrounding grid cells and those with high traffic, the neighbor cells correspond to those cells that surround the center-targeted [93]. The surrounding grid cells with the highest traffic are those with the highest correlation with the center-targeted and are defined based on statistical analysis and taking into account that these cells should correspond to regions areas associated with a high risk of accidents [51].

6.3.2.2 Transformer Architecture

Several ML techniques [93] can be applied to the produced spatiotemporal grid-based traffic sequences, enhanced by additional features, to predict future vessel traffic. In this study, we utilize the Transformers architecture, harnessing its ability to capture long-term dependencies in sequential data. For the VTFF problem, we adapt the Transformer model [95] to sequence-to-sequence forecasting. The model architecture consists of an encoder layer, a Self-Attention Mechanism, Feed-forward NNs, and a Decoder with Linear Layer as presented in Figure 18.

Particularly, a brief description of each employed layer is given below:

- **Encoder Layer:** The encoder layer processes the input sequence. For the VTFF application, this layer learns to understand and encode the patterns in the historical time series data of each grid cell.
- **Self-Attention Mechanism:** Self-attention allows the model to weigh the importance of different points in the time series. For forecasting problems, this is particularly useful as it helps the model to focus on the most relevant parts of the past data when predicting future values.
- **Feed-forward NNs:** These networks, placed within each Transformer block, allow for the learning of complex relationships. In the context of VTFF application, they can help in capturing non-linear dependencies, which are often present in maritime data. Our solution includes 4 layers of Self-Attention and Feed-forward NNs.
- **Decoder with Linear Layer:** The decoder is responsible for generating the output sequence. In the VTFF case, this represents the future values of the time series. The linear layer at the end of the decoder transforms the decoder's output into the forecasting future values format, configured with 6 * 2 neurons. This specific design is aligned with the goal of predicting the next 6 observations, each observation capturing 2D spatial data for a 5-minute interval. The linear layer effectively maps the decoder's output into this 6-interval forecast format.

Regarding model training, we employed Adam as the optimizer [95]. Also, we optimized the Transformers by fine-tuning the hidden layer sizes, the learning rate determining the number of epochs, and configuring the

batch size. Additionally, we implemented an early stopping procedure to prevent the network from overfitting by monitoring its performance on a validation set.

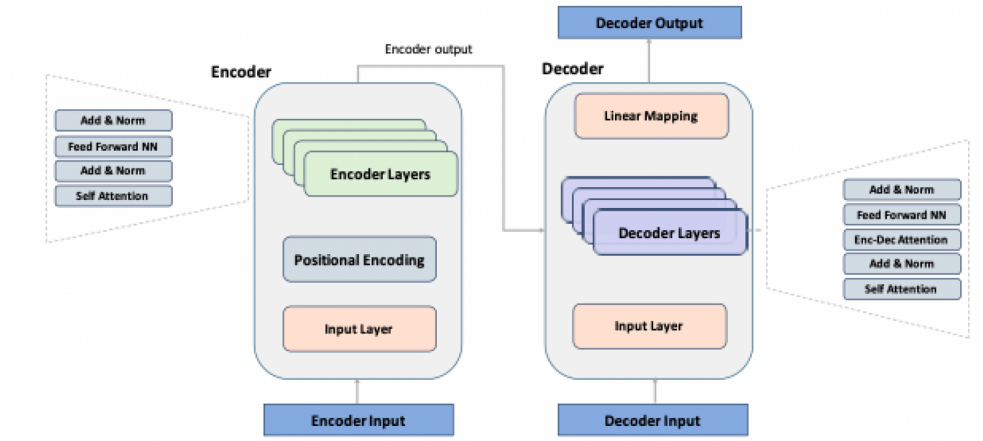


Figure 18. Network architecture, with an encoder layer, a Self-Attention Mechanism, Feed-forward NNs, and a Decoder with a Linear Layer

6.3.2.3 The distributed Unified Approach for VTFF (dUA-VTFF) method

The large amount of maritime data in the VTFF problem necessitates the deployment of a robust and scalable big data framework. The dUA-VTFF methodology harnesses the combined strengths of Apache Spark and Transformer models to ensure efficient processing of large datasets and accurate predictions. The advantage of incorporating Big Data technology and distributed processing in dUA-VTFF framework are significant gains in execution times described in the previous section and scalable architecture for maritime applications.

In our framework we leverage Apache Spark, a distributed computing system, that has the ability to process massive amounts of data seamlessly across clusters. The starting point for the Spark-based application is the establishment of a Spark session that enables the distributed processing capabilities of Spark.

The configuration of the Spark session is pivotal to the application's performance, a necessity underscored by the VTFF problem's demands for processing extensive datasets. To tailor the Spark environment to the demands of this specific problem, various parameters during the session's creation, such as the memory configuration that ensures that the available resources are used efficiently preventing out-of-memory errors and reducing garbage collection.

The distributed version of the UA-VTFF, namely dUA-VTFF pipeline is illustrated in Fig. 18. The dUA-VTFF starts by giving the available vessels' historical trajectories into the proposed VRF-LSTM algorithm to predict the future trajectories. Subsequently, the vessels' historical and future trajectories data are consolidated and converted into RDDs, laying the groundwork for efficient distributed processing across the computational framework, i.e. data preprocessing is conducted using Spark's RDD mechanism. RDD is the primary concept that Spark offers, and it is a collection of elements partitioned across the cluster nodes that can be operated on in parallel [110].

A crucial step in this process is the formation of a spatiotemporal grid constructed by aggregating the RDDs based on time and space dimensions. The available data (the vessels' historical and future trajectories), once pre-processed, are assigned into this spatiotemporal grid. Specifically, the available data are partitioned based on the unique spatial grid cells, allowing each to be treated as an independent unit for training.

This dual segmentation facilitates a more granular and precise analysis of traffic flow patterns. Following the grid creation, the calculation of the number of vessels (in the form of RDDs) that are placed in every grid cell is performed. Then, each RDD representing traffic flows undergoes a comprehensive feature vector analysis. This analysis is pivotal in generating a rich set of features for each grid cell. The features encapsulate various aspects, including the influence of neighbouring grid cells, historical traffic patterns, and seasonality aspects. This multi-faceted approach ensures a thorough understanding of the dynamics at play within each grid.

The final phase of the process involves a distributed approach to VTFF prediction. Each node in the distributed system is allocated a specific subset of grid cells. Within each node, the process unfolds in two stages for each grid cell. Initially, a top feature selection is conducted to identify the most relevant features that influence traffic flow and for every grid cell b the produced traffic sequences are enhanced with additional features. Subsequently, Transformer models are applied to these enhanced traffic sequences for VTFF prediction. This two-stage process, conducted in parallel across nodes, leverages the distributed nature of our framework, ensuring efficient and accurate forecasting.

The Transformer models, also leverage Spark's distributed features for training. The framework is designed to auto-detect available GPUs, distributing the computational workload of the model across them. This framework trains the Transformer model on each unique spatial grid cell simultaneously across a cluster, which is a component of Spark's RDD parallelization. This distributed approach ensures the framework's scalability, guaranteeing its readiness for ever-growing maritime data.

6.4 Experimental Setup

This section presents the experimental setup for evaluating dUA-VTFF method including various learning schemes for comparison.

6.4.1 Dataset description

Experiments were conducted using real-world AIS data collected from two distinct sea areas to assess the performance of the implemented models across various motion patterns.

The datasets used in this chapter were introduced in Section 2.3.1. In particular, the experimental evaluation relies on the Aegean-Cyclades and Brest AIS datasets, which provide two complementary maritime traffic environments for assessing the behavior of the proposed distributed unified framework under realistic conditions. Since their general characteristics have already been summarized in Chapter 2, the present section focuses only on the dataset properties that are directly relevant to large-scale forecasting, distributed processing, and the experimental configuration of dUA-VTFF.

In the context of this chapter, the Aegean-Cyclades dataset is used to evaluate the proposed framework in a maritime region with heterogeneous regional traffic patterns and realistic route variability, while the Brest dataset serves as a complementary benchmark for examining whether the observed methodological behavior remains stable across a distinct traffic regime. The use of two different AIS datasets is important because it allows the chapter to assess not only predictive performance, but also the robustness of the framework across different operational settings.

The chapter-specific preprocessing follows the common logic of the earlier forecasting chapters, but here the emphasis is on supporting scalable data preparation and grid-based learning. Raw AIS records are transformed into structured spatiotemporal inputs suitable for both route-level and flow-level modeling, and the resulting data representations are organized so as to support distributed computation and repeated large-scale experimentation. Thus, the role of the datasets in this chapter is not only to provide a forecasting benchmark, but also to stress the computational feasibility of the proposed framework under realistic data size.

6.4.2 Experimental setup

In this section, we present the experimental setup employed in our study. We provide insights into the hardware and software setup, along with the evaluation criteria adopted to assess the efficacy of the implemented algorithms. Furthermore, we detail the spatiotemporal grid configuration, and outline the time series generation process for analysis. Finally, we discuss the dataset splitting, as well as the model parameterization utilized in our experiments.

Hardware and Software Environment

All the methods were implemented using Python and the experiments were carried out on a workstation equipped with an Apple M1 Pro 10-Core CPU of 16 GB of RAM, and a 12-Core GPU. Also, PySpark was used for implementing dUA-VTFF, while the Transformer models were designed in PyTorch.

Performance Metric

Relative Absolute Error (RAE) [112], described by the below equation, was adopted for evaluating the performance of the models. RAE is calculated by taking into account the total absolute error and dividing it by the absolute difference between the mean and the actual value:

$$RAE = \sum_{i=1}^{N} \frac{|y_i - \hat{z}_i|}{|y_i - \bar{y}|} \quad (13)$$

with different values of spatial grid's resolution, number of temporal transitions and prediction horizon. More information can be found in [93].

Statistics & Time Series Generation

The traffic flow generated using the Aegean-Cyclades dataset and the abovementioned parameters is summarized as follows: A total of 14,340 cells were created, with traffic flow values ranging from a minimum of 0 to a maximum of 103,617 vessels. Also, the median and mean traffic flow was 25 and 120 vessels, respectively. Out of all the grid cells, only 768 cells exhibited a traffic flow exceeding 300 vessels throughout the entire observation period, while 4,000 cells had fewer than 10 vessels during the entire duration. The top 10 grid cells, on average, had more than 5 vessels every 5 minutes and included more than 300,000 vessels during the entire period.

The traffic flow generated using the Brest dataset and the abovementioned parameters is summarized as follows: A number of 17,693 cells were created, with the traffic flow of min, max, median and mean values equal to 1, 491478, 2, and 96 vessels, respectively. Of the whole grid cells, only 363 cells presented traffic flow of more than 300 vessels across the entire period, while 13,747 cells included less than 10 vessels in the whole period. The top 10 grid cells have more than 3 vessels on average every 5 minutes and include more than 1,000,000 vessels in the entire period.

As a result, for each maritime dataset, we generated 10 distinct chronologically ordered timeseries, each of the same length, corresponding to traffic volume. These time series were derived from the 10 busy grid cells and were sampled at a rate of 5 minutes. We selected a 5 min sampling rate, as it is commonly used in vessel

traffic flow applications [59, 113]. Previous studies have also indicated that using a 5-minute traffic interval leads to more accurate predictions [103].

Dataset Splitting

The traffic volume records of each timeseries were, also, allocated into three sets, namely training, validation, and testing, using a 70%–5%–25% percent ratio, respectively. Particularly, due to the time dependence, we followed a chronological order to split into the three datasets. The training set was used to define the parameters for the models, while the validation set was used to choose the model. The performance of the selected model was evaluated on the testing set. Aside from training and validation sets, it is crucial to evaluate an ML model on an independent set due to overfitting that could occur with respect to the validation set.

Model Parameterization

Regarding the model parameterization, several aspects were taken into account and optimized accordingly. More specifically, the proposed transformer models (described in section 4.2.2) were optimised by using Adam algorithm and according to the following parameters:

- **Learning Rate:** Controls the size of the steps that the model takes during the optimisation process. Smaller values lead to smaller adjustments to the model which requires more iterations in order to have convergence. We explored values ranging from 0.001 to 0.01 in our grid search.
- **Number of Layers:** Each layer consists of self-attention and Feed-forward NNs, as illustrated in Fig. 18 depicts the model's architecture. The more the layers the higher the complexity of the model. We explored values ranging from 1-12 layers in our grid search.
- **Dropout parameter:** This parameter addresses the regularization of the algorithm and helps prevent overfitting. During training, a fraction of the nodes (set by the dropout rate) are randomly dropped, forcing the model to learn redundant representations and making it less sensitive to specific features of the training data. The higher the values the higher is the amount of regularization. We explored values ranging from 0.1 to 0.4 in our grid search.
- **Batch Size:** Represents the number of training observations that are used in one iteration of the models training. The larger the batch size the higher the requirements of memory and computational power. We explored smaller values ranging from 64 - 256 and in larger values ranging from 1000 - 3000.
- **Number of heads:** Determines the number of parallel attention heads in each layer. By increasing this parameter, the model can attend to different parts of the input sequence, potentially capturing a richer set of patterns. We iterated on a range of 4 -16 heads.
- **Number of Epochs:** An epoch represents a complete pass over the entire training dataset. We iterated on a range of 30-200 epochs.

6.4.2.1 Comparison to state-of-the-art methods

To assess the effectiveness of the proposed dUA-VTFF method compared to stateof- the-art methodologies, we implemented the approaches introduced in [102,103]. The former work marked a significant milestone in the history of deep learning-based shortterm traffic forecasting [114]. Both methods in [102, 103] have been employed as baseline benchmarks in recent studies [115, 116], and [117, 118], respectively. Also, both studies [102,103] applied their respective methods to the Caltrans Performance

Measurement System (PeMS) database [102]. This dataset contains traffic flow data collected from sensors installed on freeways in California. Note that a constant sampling rate of 5 min is employed in both studies [102,103].

In the case of [102], traffic data from different sensors were aggregated to calculate the average traffic flow volume for each freeway. Thus, the traffic volume data for each freeway constitutes a distinct timeseries. Subsequently, SAEs were applied to historical traffic volume data recorded at previous time intervals in each freeway, to predict the traffic flow of each freeway at a future time interval. The output dimension of the SAEs matched the number of freeways, and the input dimension was calculated as the output dimension multiplied by the number of previous time intervals. In this study, the traffic volume data of each freeway in the PeMS dataset corresponds to the traffic volume measured in each grid cell within the maritime dataset's sea area.

On the other hand, in [103], the authors considered only the traffic volume measured by each sensor, without taking into account the freeways. Consequently, the traffic volume data collected by each distinct sensor constituted a separate timeseries. Subsequently, a unique NN model was created for each of these traffic timeseries. Specifically, for each sensor GRU models were applied to historical traffic volume data recorded at previous time intervals to predict the traffic flow for that specific sensor at a future time interval. The output dimension of these models was set to one, and the input dimension was determined by the number of previous time intervals. In this study, the traffic volume data of each sensor in the PeMS dataset corresponds to the traffic volume measured in each grid cell within the maritime dataset's sea area.

Furthermore, in this study, the parameters for the SAE and GRU models were selected according to [102,103], respectively. Specifically, the number of previous time intervals was set equal to 6, and the SAES' output dimension was configured to match the number of the top busy grid cells, which is equal to 10.

6.4.2.2 Big data distributed processing framework

To validate the efficiency of the dUA-VTFF method, a series of experiments were conducted aimed at benchmarking the framework's responsiveness to varying computational demands. The primary objective was to determine if the framework could dynamically scale and optimize its resources to meet the growing data processing requirements, a core attribute of big data solutions. These experiments conducted on the Aegean - Cyclades dataset.

More specifically, the first experiment focused on preliminary scalability testing conducted on a single node, where we incrementally increased the number of worker nodes involved in the data processing task. Specifically, the number of workers nodes took the values of 2, 4, 8, and 16. This setup simulated a real-world scenario where computational resources, such as CPU cores, GPU cores, or entire servers, can be dynamically added or removed based on the task demands. However, it's important to recognize that scalability in a distributed system could differ significantly from that of a single node.

Furthermore, the second experiment explored the framework's capability to handle vast amounts of data, a key feature of big data challenges. We progressively increased the dataset's size and assessed the framework's performance in terms of processing speed and efficiency.

Finally, the execution time for the dUA-VTFF and UA-VTFF methods was calculated in the Aegean-Cyclades dataset. This comprehensive analysis of execution times for both distributed and centralized approaches allows us to assess and compare the efficiency and performance of each method. Understanding the computational demands of these methods is crucial for making informed decisions regarding their applicability in real-world scenarios.

6.4.3 Results

This section presents the results produced by the proposed dUA-VTFF method, along with a comparative analysis involving other learning schemes.

6.4.3.1 Evaluation of Models' Prediction Accuracy

Results for the two available datasets are summarized in Table 11 and Table 12, which depict the RAE metric between the original and the predicted number of vessels per grid cell and prediction time horizon. These results are provided for the testing set, for the top 10 grid cells, covering all the implemented methods.

Method	**Time prediction horizon (min)**					
	5	10	15	20	25	30
dUA-VTFF	**0.5809**	**0.6278**	**0.7118**	**0.7897**	0.8090	**0.7916**
UA-VTFF (LSTM)	0.7082	0.7058	0.7538	0.8162	0.8236	0.8220
GRU	0.6738	0.6635	0.7516	0.7952	**0.8054**	0.7957
SAE	0.7271	0.7210	0.7833	0.8692	0.8993	0.8976

Table 11. Prediction results (RAE) for different methods in the Aegean-Cyclades dataset

Method	**Time prediction horizon (min)**					
	5	10	15	20	25	30
dUA-VTFF	**0.3043**	**0.3810**	**0.3978**	**0.4448**	**0.4655**	**0.5065**
UA-VTFF (LSTM)	0.3318	0.4110	0.4441	0.4782	0.5264	0.5511
GRU	0.3208	0.4053	0.4076	0.4556	0.4744	0.5073
SAE	0.3487	0.4128	0.4328	0.4687	0.5291	0.5599

Table 12. Prediction results (RAE) for different methods in the Brest dataset.

The proposed method dUA-VTFF consistently outperforms the other methods in terms of RAE across all prediction horizons in both datasets, except for the Aegean-Cyclades at the 25-min. prediction horizon. The effectiveness of dUA-VTFF is attributed to the carefully designed and proposed methodology, which relies on the utilization of Transformer neural networks. The attention mechanisms inherent in Transformers allow the model to capture intricate spatial and temporal dependencies, enhancing its predictive capabilities. This is particularly evident in shorter time horizons (5 to 15 min), where dUA-VTFF exhibits a significant performance advantage.

On the other hand, the GRU model [103] consistently outperforms the UA-VTFF (LSTM) [93]. This observation sheds light on the effectiveness of GRUs in the given context, possibly due to their ability to retain relevant information while mitigating vanishing gradient problems. Also, these results align with those reported in [102]. Additionally, the SAE method [102] yields the least favorable results, indicating limitations in its ability to capture the complexity of vessel dynamics in these scenarios compared to the other implemented techniques.

A critical consideration is the inherent differences in Transformer, LSTM, GRU, and SAE models. Transformers, with their self-attention mechanism, excel in capturing contextual dependencies across the entire sequence, showcasing their effectiveness in handling both sequential and spatial relationships. LSTM models, known for their memory cell and gating mechanisms, are proficient in capturing long-term dependencies but may face challenges in certain spatiotemporal contexts. GRUs, similar to LSTMs, offer advantages in capturing temporal dependencies but with a simplified structure, often proving effective in various sequence

modelling tasks. In contrast, the SAE model relies on unsupervised learning through layer-wise pre-training and subsequent fine-tuning. While this method has shown success in certain applications, its inferior performance in our context suggests limitations in capturing the complexities of spatiotemporal vessel patterns. As the prediction horizon extends from 20 to 30 min, dUA-VTFF's performance metrics become more comparable with those of the GRU model, suggesting considerations for task-specific time horizons. This shift emphasizes the importance of selecting models based on the specific characteristics and requirements of the prediction task. This implies a complex relationship between prediction accuracy and time horizon, necessitating further investigation into model behavior as temporal spans increase.

To assess the statistical significance of the performance differences observed among the various methodologies employed across the top 20 grids, we conducted a comprehensive statistical evaluation. Specifically, we compared the performance of dUA-VTFF against that of three alternative approaches: UA-VTFF, GRU, and SAE.

Method compared with dUA-VTFF	T-statistic	p-value
UA-VTFF	-3.6752	0.0383
GRU	-3.3671	0.0442
SAE	-8.0712	0.0003

Table 13. Statistical significance tests for the Aegean-Cyclades dataset

Table 13 presents the results of the conducted statistical significance analysis on the Aegean-Cyclades dataset.

As a preliminary step in our analysis, we conducted the Shapiro-Wilk test for normality on the performance differences observed between dUA-VTFF and each of the other methodologies. This test was chosen for its robustness in assessing the assumption of normal distribution in small to medium-sized datasets. Remarkably, in all three instances, the test failed to reject the null hypothesis, indicating that the differences in performance metrics were normally distributed.

Given the confirmation of normal distribution, we proceeded with paired sample t-tests to directly compare the performance of dUA-VTFF against each of the other three methodologies. This statistical test was selected for its suitability in comparing the means of two related groups, where each observation in one group is uniquely paired with a corresponding observation in the other group. Such a pairing is particularly relevant in our context, where each grid serves as a unique scenario under which the performances of two methodologies are compared.

Across all comparisons -dUA-VTFF vs. UA-VTFF, dUA-VTFF vs. GRU, and dUA-VTFF vs. SAE-we observed statistically significant differences at the 0.05 significance level. In summary, the success of dUA-VTFF in surpassing other models lies in its ability to leverage Transformer NNs, offering improved accuracy in capturing complex spatiotemporal patterns, particularly in shorter prediction horizons. The contrasting performance of GRU, LSTM, and SAE models provides valuable insights into their suitability for this specific predictive task, contributing to a comprehensive understanding of their respective strengths and limitations.

6.4.3.2 Evaluation of Models' Efficiency and Scalability

In relation to the scalability experiment, which involves investigating the framework's response to an increasing number of worker nodes, the results from Fig. 19a provide evidence of the system's scalability.

However, it should be noted that preliminary scalability testing was conducted on a single node, whereas the scalability on a distributed system could be significantly different.

Starting with a single worker node, the processing time stood at 500 minutes. As the number of nodes doubled from 1 to 2, there was a noticeable reduction in processing time to 351 minutes. Further doubling the nodes to 4 led to a decrease in time to 261 minutes. This trend continued, with 8 nodes processing the data in 210 minutes and 16 nodes further reducing the time to 180 minutes. The diminishing returns on time saved as nodes increased confirmed the distributed framework's efficiency. In summary, initial results showed a significant decrease in processing times with an increase in the number of worker nodes, confirming the framework's ability to distribute the workload effectively and leverage parallel processing.

In relation to the dataset size experiment, which involves assessing the framework's performance with progressively larger datasets, the results from Fig.19b showcases the robustness of the dUA-VTFF method. A dataset size of 5 grids was processed in 320 minutes. However, as the dataset size doubled to 10 grids, the processing time also expanded to 500 minutes. This pattern of increasing processing times continued consistently with the growth of dataset sizes. Despite the inevitable increases in processing time with larger datasets, the framework exhibited resilience and consistent performance throughout. As anticipated, the distributed framework demonstrated its strength in managing large datasets resulting in consistent and efficient performance even with growing data sizes.

Furthermore, Table 14 presents the execution time in seconds for the dUA-VTFF and UA-VTFF methods in the Aegean dataset for the top 5 and 20 grids using 150 epochs. The dUA-VTFF approach demonstrates a gain in execution time of approximately 40% compared to the UA-VTFF technique.

In conclusion, these experiments highlighted the fundamental advantages of the big data distributed framework in addressing the VTFF problem. The dUA-VTFF method's competence in a distributed big data environment is evident since it not only scales effectively with increased computational resources but also manages vast data sizes with consistent performance.

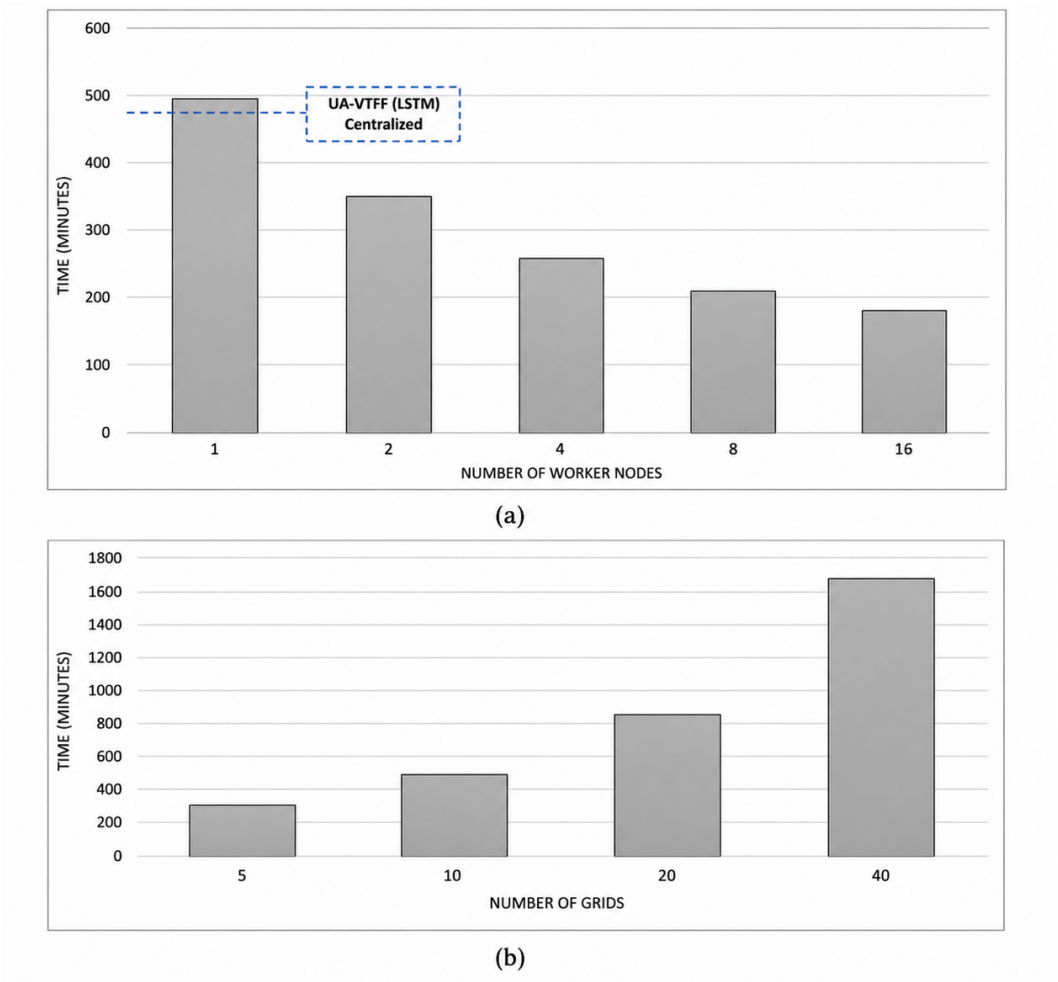


Figure 19. dUA-VTFF efficiency experiments in the Aegean-Cyclades dataset: (a) Scalability, (b) Increased data size

Method	Time (sec)	
	Top 5 Grids (150 Epochs)	Top 20 Grids (150 Epochs)
dUA-VTFF	1252	4904
UA-VTFF	2086	8135

Table 14. Execution time in the Aegean-Cyclades dataset

6.5 Discussion

This chapter extends the unified forecasting direction of the thesis by showing that predictive integration alone is not sufficient when mobility data become larger and forecasting pipelines become more computationally demanding. Chapter 5 demonstrated that vessel-level and flow-level predictive information can be combined in a unified architecture. The present chapter adds a second and equally important dimension: such a unified architecture must also remain operationally feasible under large-scale AIS data conditions.

A first important observation is that the contribution of dUA-VTFF is not limited to a new model architecture. Its deeper significance lies in the fact that it treats scalability as part of the forecasting problem itself. In many mobility-analytics studies, scalability is discussed only as an engineering concern that follows once the predictive model has been designed. The results of this chapter suggest a different view. In big

mobility data settings, the practical usefulness of a forecasting method depends not only on its accuracy, but also on whether preprocessing, feature construction, training, and inference remain feasible as data volume and model complexity increase.

A second observation concerns the relationship between unified forecasting and computational design. The unified approach of Chapter 5 already indicated that aggregate traffic prediction can benefit from combining vessel-level and flow-level predictive information. However, the integration of these two information sources inevitably increases the complexity of the forecasting pipeline. The present chapter shows that this added complexity must be matched by an architecture that is explicitly designed for scale. In this sense, dUA-VTFF should be understood not as a simple replacement of UA-VTFF, but as its large-scale continuation.

A third observation is that the Transformer-based perspective becomes especially relevant at this stage of the thesis. Earlier chapters focused mainly on benchmarking and on the integration of predictive signals. In the present chapter, the forecasting problem is also treated as a representation-learning problem over structured spatiotemporal inputs. The Transformer-based design is useful in this context because it provides a flexible mechanism for capturing complex temporal dependencies while remaining compatible with larger-scale processing pipelines. At the same time, the chapter makes clear that architectural power alone is not enough; the surrounding distributed pipeline is equally important for achieving usable large-scale forecasting.

A fourth observation concerns the trade-off between predictive quality and computational feasibility. A forecasting method that is only marginally more accurate but substantially more expensive may not always be operationally preferable. Conversely, a scalable method that sacrifices too much predictive quality may also be of limited value. The contribution of dUA-VTFF lies precisely in studying these two dimensions together rather than in isolation. This is one of the most important thesis-level messages of the chapter: in big mobility data, forecasting quality and system feasibility must be analyzed jointly.

A fifth observation is that the use of multiple datasets strengthens the methodological argument of the chapter. By evaluating the framework on both Aegean-Cyclades and Brest, the chapter moves beyond a single maritime case study and tests whether the observed gains are stable across distinct traffic environments. This is important because scalability claims are meaningful only when they are supported under more than one data regime. In that sense, the chapter contributes not only a scalable forecasting method, but also a more robust evaluation perspective.

The chapter also has limitations. Although the proposed framework addresses important scalability issues, the experiments are still confined to specific AIS datasets and to a particular distributed design. Other distributed architectures, alternative spatial representations, or different ways of integrating route-level and flow-level information may lead to different trade-offs. In addition, as the system becomes more sophisticated, interpretability may become more difficult than in simpler direct or indirect models. These are natural limitations of moving toward richer large-scale forecasting systems.

Within the broader structure of the thesis, Chapter 6 is important because it completes the progression from benchmarking, to comparison, to integration, and finally to scalable integration. Chapter 3 established a benchmark for individual route forecasting. Chapter 4 compared the two principal aggregate forecasting paradigms. Chapter 5 showed how those paradigms can be combined. Chapter 6 now demonstrates that this combination can be extended into a distributed and computationally feasible framework suitable for larger data settings. This progression is central to the dissertation's claim that data mining in big mobility data must be treated as both a modeling problem and a systems problem.

Finally, the chapter prepares the transition to Chapter 7. After examining predictive integration and computational scalability in the maritime forecasting setting, the thesis turns to a more methodological deep-

learning contribution, namely trainable activation functions. In this sense, Chapter 6 closes the large-scale forecasting part of the thesis and opens the way to the more architecture-oriented methodological investigation that follows.

6.6 Summary

This chapter presented dUA-VTFF, a distributed and Transformer-based extension of the unified forecasting direction introduced in Chapter 5. The aim of the chapter was not only to improve aggregate vessel traffic flow prediction, but also to ensure that unified forecasting remains computationally feasible when applied to larger AIS datasets and more demanding experimental settings.

To achieve this, the chapter reformulated unified forecasting within a distributed processing perspective and combined route-level and flow-level predictive information in a scalable architecture. In this way, the chapter extended the contribution of UA-VTFF from predictive integration alone to predictive integration under explicit computational constraints. This is an important step in the overall thesis progression, because it shows that forecasting in big mobility data must be evaluated not only by its predictive accuracy, but also by its feasibility at scale.

The results showed that dUA-VTFF provides a meaningful large-scale continuation of the unified forecasting paradigm. Rather than treating scalability as an implementation detail, the chapter demonstrated that data preparation, representation, learning architecture, and computational design interact directly with forecasting performance. This reinforces one of the central claims of the dissertation: in mobility analytics, the design of forecasting systems must address both methodological rigor and operational scalability.

An additional contribution of the chapter is that it broadens the evaluation perspective of the thesis. By using more than one AIS dataset and by studying predictive quality together with computational behavior, the chapter supports a more realistic view of what large-scale mobility forecasting requires in practice. This makes Chapter 6 a key point of consolidation for the forecasting part of the dissertation.

Within the broader structure of the thesis, Chapter 6 completes the transition from isolated forecasting paradigms to scalable unified forecasting architectures. It therefore serves as the final chapter of the main forecasting arc before the dissertation turns to a distinct but related methodological contribution in deep learning.

The next chapter moves away from the direct study of forecasting pipelines and examines trainable activation functions as reusable neural components that can improve model flexibility across forecasting settings and beyond. In this sense, Chapter 6 closes the scalable forecasting part of the thesis and prepares the transition to the methodological deep-learning contribution of Chapter 7.

The material presented in this chapter is based on and extends the publication “A transformer-based method for vessel traffic flow forecasting” [84].

Part IV. Methodological Advances in Deep Learning

Chapter 7: KANB – Trainable Parametric Activation Functions

7.1 Introduction

The previous chapters focused on mobility forecasting under increasingly realistic and large-scale settings, culminating in distributed Transformer-based traffic flow forecasting. We now turn to a different but complementary research direction: rather than proposing a new forecasting architecture, this chapter studies a methodological component of neural networks themselves, namely the activation function. This shift is important because forecasting performance is influenced not only by the global model design, but also by the nonlinear transformations through which signals are propagated and learned.

Artificial Neural Networks (ANNs) have become central to modern machine learning because they can learn complex nonlinear mappings from large datasets and have shown strong performance across time series forecasting, computer vision, and other domains [121], [122]. At the same time, their effectiveness depends heavily on both trainable parameters and researcher-defined hyperparameters. Among the latter, activation functions are especially important because they determine how nonlinear behavior is introduced into the network and therefore affect gradient propagation, convergence speed, representational flexibility, and final predictive performance [124]–[127].

A broad range of activation functions has been proposed in the literature, including classical nonlinearities such as Sigmoid, Tanh, Softplus, ReLU, LeakyReLU, ELU, Swish, and Mish [129]–[134]. More recently, adaptive and trainable activation functions have attracted growing interest because they allow the nonlinearity itself to be learned rather than fixed a priori [123], [136]–[141]. This is particularly relevant in forecasting tasks, where different data regimes may require different nonlinear responses during training.

Motivated by this line of work, this chapter introduces KANB, a family of parametric activation functions based on cubic Bézier curves. The central idea is to use a flexible curve-based formulation with trainable control points so that the activation shape can adapt during learning. In this way, the chapter contributes not another forecasting architecture, but a modular neural component that can be inserted into different deep-learning pipelines.

To evaluate this contribution, KANB is integrated into the dUA-VTFF forecasting framework in order to assess its effect on mobility forecasting under realistic spatiotemporal conditions.

7.2 Problem Formulation

Bézier curves [142] use control points in their parametric equations making them easy to manipulate regarding monotonicity and curvature. A cubic Bézier that passes through the origin and point (1,1) can be defined as the parametric curve:

$$B(t) = 3(1-t)^2\, t\, P_1 + 3(1-t)\, t^2\, P_2 + (1,1)\, t^3 \tag{14}$$

where $t \in [0, 1]$ is the parameter of the curve and $P_1 = (X_{P_1}, Y_{P_1})$, $P_2 = (X_{P_2}, Y_{P_2})$ are the control points.

In this section, a new set of parametric activation functions is introduced. The general form of KANB activations can be expressed as:

$$KANB(y) = 3\big(1 - sigmoid(y)\big)^2 sigmoid(y) Y\, P_1$$

$$+ 3\big(1 - sigmoid(y)\big)\big(sigmoid(y)\big)^2 Y\,P_2 \; + \; g(y)\big(sigmoid(y)\big)^3 + \; f(y) \tag{15}$$

where Y_{P_1}, Y_{P_2} are trainable parameters, bounded between [0, 1], f(x) is a function that varies for each KANB variant and g(y) = 1 or g(y) = y depending on the choice of endpoints. The use of sigmoid serves the purpose of having sigmoid(y) taking values in the interval [0, 1].

KANB1 is the ordinate of the cubic Bézier curve (19), which is a Bernstein polynomial, with the parameter of the curve being the sigmoid of $y = \sum w_i x_i + \; b_i$.

That is,

$$\begin{aligned} KANB1(y) = \; & 3\big(1 - sigmoid(y)\big)^2 sigmoid(y) Y\,P_1 \\ & + 3\big(1 - sigmoid(y)\big)\big(sigmoid(y)\big)^2 Y\,P_2 \\ & + \big(sigmoid(y)\big)^3 \end{aligned} \tag{16}$$

In KANB1 Y_{p1} , Y_{p2} can be considered as trainable parameters, the proper choice of which can yield a more suitable activation function. Taking into account that the derivative of the sigmoid is given by

$$\frac{d\big(sigmoid(y)\big)}{y} = \; sigmoid(y) \cdot \big(1 - \; sigmoid(y)\big)$$

it follows that the derivative of KANB1 can be expressed as

$$\begin{aligned} \frac{d(KANB1)}{dy}(y) = \; & 3\big(1 - \; sigmoid(y)\big) sigmoid(y) \cdot \Big[\, YP_1\big(1 - \; sigmoid(y)\big)\big(1 - \; 3\, sigmoid(y)\big) \; - \\ & YP_2\Big[\big(sigmoid(y)\big)^2 - \; 2\big(1 - \; sigmoid(y)\big) sigmoid(y)\Big] + \; \big(sigmoid(y)\big)^2\Big] \end{aligned}$$

KANB1 is saturating since

$$\lim_{y \to \infty} \Big(\frac{d(KANB1)}{dy}\Big) = \; 0 \tag{17}$$

In order to mitigate the possibility of vanishing gradients KANB2 is introduced. KANB2 is the ordinate of a cubic Bézier with endpoints (0,0) and (y,y), i.e.:

$$\begin{aligned} KANB2\,(y) = \; & 3\,\big(1 - \; sigmoid\,(y)\big) 2\; sigmoid\,(y) YP_1 \\ & + 3\,\big(1 - \; sigmoid\,(y)\big)\big(sigmoid\,(y)\big)^2 YP_2 \\ & + y\;(sigmoid\,(y))3 \end{aligned} \tag{18}$$

Which has a derivative of

$$\frac{d(KANB2)}{dy}(y) = \; 3\big(1 - sigmoid(y)\big) sigmoid(y) \cdot$$

$$\Big[\, YP_1\big(1 - \; sigmoid(y)\big)\big(-\,3\; sigmoid(y)\big) - YP_2\Big[\big(sigmoid(y)\big)^2 - \; 2\big(1 - \; sigmoid(y)\big) sigmoid(y)\Big] + \; \big(sigmoid(y)\big)^2\Big] \; + \; \big(sigmoid(y)\big)^3$$

Another way to address vanishing gradients effect would be to add an extra term f(x) to KANB1. For instance adding f (x) =y we get

$$KANB3\,(y) \;=\; KANB1 \;+\; y \tag{19}$$

which is non-saturating since its derivative is

$$\frac{d(KANB3)}{dy}(y) = \frac{d(KANB1)}{dy} + 1. \tag{20}$$

We should also point out that the proper selection of f (x) may yield a more fitting activation function for a given task.

Finally, a fourth non-saturating activation function can be defined as

$$KANB4(y) \;=\; KANB1 \;+\; ln(1 \;+\; e^{y}) \tag{21}$$

The derivative of KANB4 is

$$\frac{d(KANB4)}{dy}(y) = \frac{d(KANB1)}{dy} + sigmoid(y). \tag{22}$$

It is worth mentioning that except for the derivative of KANB2, the derivatives of every other KANB activation function are polynomials of sigmoid *(y)* therefore bounded; this lessens the risk of exploding gradients. KANB2, however, is more susceptible to the effect of exploding gradients due to the y (sigmoid *(y))*^{2} term in the formula of the derivative.

Since activation functions are considered as hyperparameters of the ANN, the use of parametric activation functions allows their parameters to be updated during training in the same way as the network's weights and biases. So, members of KANB family are adaptive activation functions that update their parameters while training in the aforementioned manner. In what follows and for the sake of comparison we conducted experiments using fixed preset values for KANB parameters and afterwards treating the parameters as trainable.

7.3 Methodology

In this work, the dUA-VTFF framework is further enhanced through the integration of the proposed KANB activation functions. The motivation for this extension lies in improving the nonlinear modeling capacity of the Transformer-based components within the dUA-VTFF. While the original dUA-VTFF already achieved superior predictive accuracy in various domains through the use of attention-based temporal modeling, its reliance on fixed activation functions limited adaptability during training. By introducing KANB, a parametric and trainable activation family, we enable the framework to dynamically adjust its internal response to varying traffic flow regimes, congestion levels, and environmental influences.

The KANB-enhanced dUA-VTFF (referred to as dUA-VTFF-KANB) maintains the modular Transformer backbone but replaces all conventional activation layers with KANB variants (KANB1-4). Each activation possesses trainable control points Y_{p1}, Y_{p2} optimized through backpropagation alongside the model weights and biases. The adaptability of KANB allows the network to balance between saturating and non-saturating

behavior, yielding smoother gradient propagation and enhanced stability, particularly under long-term prediction horizons.

Mathematically, the KANB activation is applied to the linear transformations within the Transformer encoder and decoder blocks, thereby enhancing both feature extraction and temporal correlation learning. This modification results in better handling of multi-scale temporal dependencies—an essential aspect of traffic dynamics.

The dUA-VTFF - KANB Framework

We enhance the dUA-VTFF Framework with the replacement of the activation function on each layer to KANB. The trainable parameters of KANB through backpropagation alongside weights and biases. Control points are trainable parameters. Initial settings approximate known activations (e.g., ReLU, Sigmoid). During training, parameters adapt to optimize performance.

This approach introduces an additional adaptive layer within the Transformer architecture. During training, the learnable parameters of KANB evolve to approximate the most suitable nonlinear mapping for each dataset, effectively tuning the model to specific traffic flow regimes. In practice, the dUA-VTFF-KANB is initialized with KANB2 for its non-saturating properties and is fine-tuned across all KANB variants to assess generalization.

The key advantage of this enhancement is the synergistic interaction between the self-attention mechanism and the dynamic activation surface learned by KANB. Attention captures dependencies between moving object flows across spatial grids, while KANB adaptively reshapes the activation landscape, improving convergence speed and reducing prediction error in extended horizons.

In addition to its integration into the dUA-VTFF framework for traffic forecasting, the proposed KANB activation family was also incorporated into Convolutional Neural Networks (CNNs) for evaluation on computer vision benchmarks such as MNIST, CIFAR-10, CIFAR-100 and Cats vs Dogs. In these architectures, each nonlinear activation layer originally employing ReLU or Tanh was replaced by one of the KANB **v**ariants, applied after each convolutional and fully connected layer. The inclusion of trainable control points allows the convolutional filters to adapt not only their spatial feature extraction but also the nonlinear response governing feature amplification or suppression.

This dynamic adaptation enables KANB-based CNNs to better handle variations in local contrast, texture, and color composition, especially in the more complex natural image datasets such as CIFAR-10 and CIFAR-100. For simpler grayscale data like MNIST, KANB maintains comparable convergence behavior to traditional activations while demonstrating smoother gradient propagation and reduced sensitivity to initialization.

Overall, this incorporation demonstrates that KANB functions as a general activation mechanism capable of enhancing both temporal (VTFF) and spatial (vision) learning tasks, highlighting its domain-independent adaptability within deep architectures.

7.4 Experimental Setup

The datasets used in this chapter belong to the broader experimental corpus summarized in Section 2.3.1. In the present chapter, however, the main experimental emphasis is placed on mobility-related forecasting settings, since the purpose of Chapter 7 is to evaluate KANB as a methodological contribution in the context of the broader mobility-analytics focus of the thesis.

To this end, the experimental evaluation is centered on three mobility datasets that represent different forecasting environments. The Aegean-Cyclades dataset is used as the maritime forecasting setting, the Porto Taxi dataset is used as an urban-mobility setting, and the London air traffic dataset is used as an aviation-

mobility setting. This selection is important because it allows the chapter to examine the behavior of KANB across multiple mobility domains while remaining aligned with the thematic scope of the dissertation.

For each dataset, the corresponding forecasting model is trained under the same general evaluation logic used throughout the thesis, with controlled train-validation-test separation and consistent comparison across alternative activation functions. The objective is not to redesign the surrounding forecasting architecture, but to assess whether replacing standard nonlinearities with trainable KANB activations can improve predictive behavior in a modular and reusable way.

The evaluation metric used in the forecasting experiments is the Relative Average Error (RAE), introduced in Section 2.3.2. In the present chapter, RAE is used to compare the predictive behavior of forecasting models under different activation-function choices and across multiple forecasting horizons.

In addition to the mobility-oriented experiments presented in the main body of the chapter, supplementary experiments were also conducted on standard image-classification benchmarks for completeness and broader methodological validation. These additional experiments are not emphasized in the thesis body because they fall outside the central mobility focus of the dissertation. Their role is only to provide further evidence that the proposed activation family is not tied exclusively to one forecasting architecture or one application setting.

For this reason, the chapter concentrates its main analysis on the mobility datasets, where the contribution of KANB can be interpreted directly in relation to the forecasting problems studied in the rest of the thesis.

B. Neural Network Architectures

For the purposes of this work, we implemented the dUA-VTFF-KANB framework, which preserves the Transformer backbone of dUA-VTFF [84] and replaces the internal non-linearities with KANB activations.

- **Input representation:** Each dataset is represented as a spatiotemporal traffic flow tensor over a spatial grid. For each selected high-traffic cell, we build an input sequence consisting of the historical flow counts observed in the previous time steps.
- **Tokenization and embeddings:** The Transformer operates on embedded tokens. Each flow value (or flow feature vector, if additional flow descriptors are used) is projected into a latent dimension through a linear layer. To preserve temporal ordering, we add positional encodings (learned or sinusoidal) to the embedded sequence.
- **Encoder–decoder structure:** The model follows a standard Transformer encoder–decoder design.The encoder receives the embedded historical sequence and produces contextual representations through stacked self-attention blocks.The decoder predicts the future traffic flow sequence, using masked self-attention to preserve causality and encoder–decoder attention to condition on the encoded past. This design is particularly suitable for VTFF because the self-attention mechanism can learn long-range temporal dependencies and periodicities that are difficult to capture with fixed receptive fields.
- **Activation Functions:** In dUA-VTFF-KANB, we replace all conventional non-linearities inside the Transformer with KANB activations. In practice, this primarily affects the FFN sublayers, which are the main nonlinear components of Transformer blocks. Specifically:
  - The FFN activation is replaced by KANB1, KANB2, KANB3, or KANB4. When the "-opt" variant is used, the Bézier control parameters are treated as trainable parameters and optimized jointly with the network weights through backpropagation.
  - Parameter constraints are enforced via clipping to [0,1] (Clip01), ensuring the learned activation remains well-formed and stable.

- This integration is “drop-in”: no structural change is made to attention or tokenization. Therefore, any performance difference can be attributed directly to the activation function behavior.

C. TRAINING PROTOCOL

For all architectures, we implemented a consistent training protocol:

- Aegean-Cyclades: Batch size of 64
- Porto Taxi: Batch size of 32
- London Air traffic: Batch size of 32
- Validation split of 10% from training data
- Maximum training epochs of 350

This systematic approach to architecture design and training protocol ensures that any observed differences in performance can be attributed to the activation functions rather than architectural or training variations. The architectures were carefully chosen to provide sufficient capacity for each task while maintaining computational efficiency and allowing for meaningful comparison across datasets of varying complexity.

D. KANB PARAMETER OPTIMIZATION

The optimization of KANB parameters follows a systematic process:

Initialization:

- Y_{P1}: Randomly initialized between $[0.5, 0.9]$
- Y_{P2}: Randomly initialized between $[0.1, 0.4]$

These ranges are chosen to ensure the initial curve shape is reasonable

Training Process:

- Parameters are updated using backpropagation
- Gradient descent is applied to both Y_{P1} and Y_{P2}
- Parameters are constrained to $[0, 1]$ using the Clip01 constraint
- Updates occur every epoch during training

Parameter Evolution:

- The values of Y_{P1} and Y_{P2} evolve throughout training
- This evolution allows the activation function to adapt to the specific characteristics of the data and task
- The adaptation process is monitored and logged for analysis

7.5 Results

Below is a dataset-by-dataset summary of the updated experimental results, focusing on how the proposed KANB1-4 activation functions perform with and without optimized parameters, relative to traditional baselines

(ReLU, Sigmoid, Tanh). The discussion emphasizes the nuanced behavior of KANB across different datasets and training durations.

Aegean-Cyclades, in Table 15, shows a clear and consistent horizon effect: for every activation function, RAE increases from t+5 to t+30, confirming that multi-step vessel flow forecasting becomes progressively harder as the look-ahead time grows. Among the fixed baselines, ReLU is the strongest overall (mean RAE ≈ 0.44), while Sigmoid (≈ 0.47) and especially Tanh (≈ 0.505) degrade more sharply at longer horizons (e.g., at t+30: Sigmoid 0.59, Tanh 0.61, vs ReLU 0.52). This suggests that in this maritime grid setting, stronger saturation (particularly Tanh) tends to hinder long-horizon learning, likely because it limits gradient signal in regimes where vessel density patterns shift across time.

Within the KANB family without optimization, the activation choice already matters: KANB2 (mean ≈ 0.422) and KANB3 (mean ≈ 0.413) outperform ReLU on average, indicating that even fixed Bézier-shaped nonlinearities can better match the underlying dynamics of maritime traffic flows. However, the most important outcome emerges when the Bézier control points are made trainable ("-opt"). KANB3-opt achieves the lowest RAE at every horizon, improving not only short-term accuracy but also long-horizon stability.

Aegean-Cyclades

Activation Function	Time prediction horizon (min)					
	5	10	15	20	25	30
Relu	0.32	0.39	0.45	0.47	0.49	0.52
Sigmoid	0.35	0.41	0.47	0.48	0.52	0.59
Tanh	0.39	0.44	0.48	0.53	0.58	0.61
KANB1	0.33	0.40	0.44	0.48	0.51	0.53
KANB2	0.34	0.37	0.41	0.45	0.45	0.51
KANB3	0.31	0.37	0.43	0.44	0.44	0.49
KANB4	0.32	0.40	0.47	0.48	0.50	0.53
KANB1 -opt weights	0.32	0.38	0.43	0.46	0.45	0.46
KANB2 -opt weights	0.31	0.35	0.39	0.45	0.46	0.47
KANB3 -opt weights	0.30	0.33	0.37	0.39	0.42	0.44
KANB4 -opt weights	0.32	0.40	0.45	0.48	0.49	0.50

Table 15. Relative Average Error (RAE) for Aegean-Cyclades dataset for each activation function for 250 epochs and different time prediction horizon.

Porto Taxi, in Table 16, exhibits the expected horizon effect: for all activation functions, RAE increases monotonically from t+5 to t+30, confirming that long-horizon forecasting compounds uncertainty in dense urban mobility flows. Among the fixed baselines, Sigmoid performs best overall (mean RAE ≈ 0.72) compared to ReLU (≈ 0.752) and Tanh (≈ 0.798), suggesting that a smoother saturating nonlinearity is better matched to the high-variance but strongly periodic demand dynamics present in taxi traffic.

Within the KANB family without optimization, performance is mixed: KANB1–3 generally track the baselines, while KANB4 is notably strong even without trainable parameters (mean RAE ≈ 0.70), improving over ReLU and Sigmoid at most horizons. The most consistent gains arise when KANB control points are trained ("-opt"). KANB4-opt is the best method at every horizon (0.58, 0.61, 0.67, 0.67, 0.69, 0.75), and it also achieves the lowest mean RAE (≈ 0.662). Relative to ReLU, this corresponds to an average reduction of ~12.0% in error, and relative to the best fixed baseline (Sigmoid), a reduction of ~8.1%. Importantly, KANB4-

opt not only improves short-term predictions but also slows down the error growth for longer horizons, indicating better modeling of nonlinear regime shifts (bursty peaks, transitions, and the strong spatial concentration of demand in high-traffic cells).

Porto Taxi

Activation Function	Time prediction horizon (min)					
	5	10	15	20	25	30
Relu	0.63	0.7	0.74	0.78	0.81	0.85
Sigmoid	0.61	0.67	0.7	0.73	0.79	0.82
Tanh	0.67	0.73	0.79	0.84	0.87	0.89
KANB1	0.63	0.73	0.74	0.79	0.83	0.86
KANB2	0.62	0.68	0.72	0.72	0.78	0.82
KANB3	0.63	0.72	0.73	0.79	0.87	0.88
KANB4	0.6	0.65	0.7	0.7	0.76	0.79
KANB1 -opt weights	0.63	0.73	0.74	0.77	0.79	0.85
KANB2 -opt weights	0.62	0.67	0.71	0.72	0.76	0.8
KANB3 -opt weights	0.63	0.72	0.73	0.77	0.77	0.79
KANB4 -opt weights	0.58	0.61	0.67	0.67	0.69	0.75

Table 16. Relative Average Error (RAE) for Porto Taxi dataset for each activation function for 250 epochs and different time prediction horizon.

London air traffic, in Table 17, is characterized by sparse flow patterns and sharp temporal bursts, and the results reflect a clearer separation between activation choices as horizons increase. The three fixed baselines show consistent deterioration with horizon, with ReLU being the strongest of the baselines overall (mean RAE ≈ 0.54) while Sigmoid (≈ 0.56) and Tanh (≈ 0.585) degrade more at longer horizons.

In terms of overall performance, KANB2-opt achieves the lowest mean RAE (≈ 0.495), improving upon the best baseline (ReLU at ≈ 0.54) by about ~8.3% on average. The gains become especially meaningful at longer horizons (e.g., t+30: 0.57 vs 0.64 (ReLU) and 0.69 (Tanh)), indicating that the learned KANB2 nonlinearity improves stability and reduces long-horizon drift in the Transformer's predictive dynamics. Overall, London results suggest that the non-saturating formulation of KANB2 combined with trainable control points is particularly effective in sparse, bursty flow settings, where maintaining gradient signal and adapting the activation curvature helps the model track intermittent peaks more reliably.

London air traffic

Activation Function	Time prediction horizon (min)					
	5	10	15	20	25	30
Relu	0.44	0.50	0.53	0.53	0.6	0.64
Sigmoid	0.46	0.51	0.53	0.57	0.62	0.67
Tanh	0.45	0.53	0.59	0.61	0.64	0.69
KANB1	0.43	0.49	0.51	0.55	0.57	0.63
KANB2	0.41	0.46	0.51	0.52	0.55	0.59
KANB3	0.42	0.48	0.52	0.56	0.57	0.6
KANB4	0.43	0.49	0.51	0.55	0.57	0.64
KANB1 -opt weights	0.43	0.49	0.51	0.55	0.55	0.6
KANB2 -opt weights	0.41	0.44	0.5	0.52	0.53	0.57
KANB3 -opt weights	0.42	0.48	0.52	0.52	0.55	0.59
KANB4 -opt weights	0.43	0.48	0.51	0.55	0.57	0.61

Table 17. Relative Average Error (RAE) for London air traffic dataset for each activation function for 250 epochs and different time prediction horizon.

Although the main body of the chapter focuses on mobility forecasting datasets, additional experiments were also conducted on standard image-classification benchmarks in order to assess the broader methodological behavior of KANB outside forecasting. These experiments were included for completeness and to verify that the proposed trainable activation family is not useful only in one narrow application setting.

However, because the thesis is centered on data mining in big mobility data, these non-mobility experiments are not treated as a primary empirical contribution of the thesis. Their role is supplementary: they provide supporting evidence for the general adaptability of KANB, while the main interpretation of the chapter remains grounded in the maritime, urban, and aviation forecasting settings that are directly aligned with the broader thesis theme.

Method	Time (sec)	
	Top 5 Grids (150 Epochs)	Top 20 Grids (150 Epochs)
UA - VTFF	2086	8135
dUA-VTFF	1252	4904
dUA-VTFF - KANB	1467	5264

Table 18. Execution time in the Aegean-Cyclades dataset including KANB activation functions.

Table 18 reports the end-to-end execution time of the distributed experiments for UA-VTFF, dUA-VTFF, and dUA-VTFF augmented with KANB, for two workload sizes (Top-5 and Top-20 grids) under a fixed training budget of 150 epochs. Moving from the non-distributed UA-VTFF to the distributed dUA-VTFF yields a clear reduction in runtime: from 2086 s to 1252 s for Top-5 grids (≈40% faster) and from 8135 s to 4904 s for Top-20 grids (≈40% faster), confirming the benefit of distributed processing as the number of grids increases. When KANB is introduced on top of dUA-VTFF, runtime increases relative to dUA-VTFF alone

(1467 s vs. 1252 s for Top-5; 5264 s vs. 4904 s for Top-20), reflecting the additional computation introduced by KANB; however, the method remains substantially faster than UA-VTFF (≈30% faster for Top-5 and ≈35% faster for Top-20). Importantly, this moderate overhead is justified by the improved predictive performance observed with dUA-VTFF-KANB in the accuracy results, indicating that the added modeling capacity of KANB trades a small increase in runtime for more accurate forecasts while retaining most of the scalability benefits of the distributed pipeline.

7.6 Discussion

The findings of this chapter indicate that KANB constitutes a meaningful methodological contribution to neural forecasting models. While the previous chapters focused on forecasting paradigms, unified architectures, and scalability under large mobility datasets, the present chapter examined a different but complementary dimension of the learning process, namely the role of the activation function as an internal component that influences optimization behavior, representational flexibility, and predictive performance.

A first important observation is that activation functions should not be treated as neutral design choices. In forecasting models operating on noisy, irregular, and heterogeneous mobility data, the form of the nonlinearity can affect gradient propagation, convergence speed, and the capacity of the model to adapt to different temporal patterns. The results suggest that allowing the activation function to adapt through trainable parameters can improve the flexibility of the model without requiring changes to the overall forecasting architecture.

A second observation is that the value of KANB lies in its modularity. The proposed activation family is not introduced as a replacement for the forecasting models developed in the earlier chapters, but as a reusable neural component that can be incorporated into different architectures. This makes the contribution methodologically broader than a single task-specific model. In practical terms, it means that improvements in forecasting behavior can emerge not only from redesigning the forecasting pipeline, but also from refining the internal nonlinear transformations used by the models.

A third observation concerns the consistency of the results across mobility-related forecasting settings. By evaluating KANB on maritime, urban, and aviation datasets, the chapter shows that the proposed activation functions are relevant across multiple mobility domains rather than being tailored to only one data source or application scenario. This strengthens the methodological scope of the contribution and supports the broader thesis perspective that key learning challenges recur across different types of mobility data.

At the same time, the results also suggest that the benefits of trainable activation functions are not necessarily uniform across all settings. The effect of KANB depends on the interaction among the dataset, the forecasting horizon, the surrounding architecture, and the optimization process. In some cases, the gains are more pronounced, whereas in others they are more moderate. For this reason, KANB should not be interpreted as a universal replacement for all standard activation functions, but rather as a flexible alternative whose usefulness depends on the learning context.

Additional experiments were conducted on non-mobility benchmarks in order to assess the broader applicability of the proposed activation family. These experiments provide supplementary evidence that KANB is not restricted to a single forecasting domain. However, the main interpretation of the chapter remains centered on mobility-related forecasting, since this is the setting in which the contribution is most directly connected to the broader scope of the dissertation.

Overall, the chapter broadens the methodological contribution of the thesis by showing that improvements in mobility forecasting can arise not only from changes in forecasting formulation, representation, or system design, but also from more fundamental neural-network components. In this sense, KANB complements the

forecasting architectures studied in the previous chapters by providing a trainable and reusable nonlinearity that can enhance model adaptability in diverse predictive settings.

Finally, the chapter prepares the transition to the last methodological part of the dissertation. After examining forecasting-oriented methodological improvements within neural networks, the thesis now turns to the representation of trajectories and their analysis in unsupervised settings. This shift is important because it moves the discussion from prediction-oriented neural design to the broader question of how mobility data should be encoded for clustering and structure discovery.

7.7 Summary

This chapter introduced KANB, a family of trainable parametric activation functions based on cubic Bézier curves, and examined its role as a reusable methodological component in neural forecasting models. In contrast to the previous chapters, which focused on forecasting paradigms and system-level architectures, the emphasis here was placed on the internal design of neural networks and on the extent to which trainable nonlinearities can improve model behavior.

The empirical analysis concentrated primarily on mobility-related forecasting settings, including maritime, urban, and aviation datasets. Within these settings, KANB was evaluated as a modular replacement for standard activation functions, with the goal of determining whether trainable activation shapes can improve predictive performance while preserving the surrounding model architecture. This allowed the chapter to remain methodologically broad while still aligned with the central mobility focus of the dissertation.

The results showed that KANB provides a meaningful methodological extension to the thesis. Its contribution lies not in redefining the forecasting frameworks developed earlier, but in complementing them with a flexible neural component that can enhance model adaptability across different datasets and forecasting horizons. In this sense, the chapter extends the dissertation from forecasting-pipeline design to methodological robustness in deep learning.

Supplementary experiments on non-mobility benchmarks were also carried out in order to assess the broader applicability of the proposed activation family. These experiments serve as supporting evidence that KANB is not tied exclusively to one application setting. Nevertheless, the main empirical contribution of the chapter remains the study of KANB in mobility-related forecasting problems, which is the context most directly connected to the overall thesis.

Within the broader structure of the dissertation, this chapter serves as a methodological bridge between the forecasting-oriented chapters and the final chapter on trajectory representation and clustering. After addressing benchmarking, paradigm comparison, unified forecasting, scalable forecasting, and neural-model design, the thesis now turns to the question of how trajectories should be represented for unsupervised analysis. In this way, Chapter 7 closes the prediction-oriented methodological arc of the dissertation and prepares the transition to the representation-oriented contribution that follows.

Part V. Trajectory Representation and Clustering

Chapter 8: A Benchmark Framework for Trajectory Vectorization in Unsupervised Settings

8.1 Introduction

Spatiotemporal datasets are continually expanding and include videos, social media posts, GPS traces, and other time series measurements. Effective analysis of such data can have a significant societal impact, helping to address challenges in environmental and climate change, public security [143] and healthcare [152].

Several researchers in Mobility Data Science (MDS) [161] focus on understanding and analyzing this data and producing reports on various observed phenomena. This led to the development of application- and data-specific procedures [149], which reduce the reusability of algorithms. Silva et al. [147] compared various trajectory analysis methods and found that approximately 70% of them are tailored to specific applications such as transportation mode identification. To overcome this limitation, researchers have turned to the field of time series for inspiration and have explored various ways to represent spatiotemporal data to meet different analysis objectives. The concept is to transform the input sequential data into different vector-based feature spaces that can be utilized with standard, general-purpose machine learning models, thereby minimizing the need for manual feature engineering. For example, in [157], the authors explore the concept of shapelet transform, introducing GEOLET to transform input trajectories into a (dis)similarity matrix that compares the input data with specific discriminant sub-trajectories extracted from it. In this chapter, the term trajectory vectorization refers to the transformation of a trajectory, originally represented as a sequence of spatiotemporal points, into a fixed-size vector representation suitable for downstream machine learning, clustering, and similarity analysis.

Although these methods achieve state-of-the-art performance in classification tasks, they have not yet been evaluated in an unsupervised setting. We therefore propose a unified benchmarking framework for evaluating trajectory vectorization techniques in such a setting. Within this framework, we formulate the problem as trajectory clustering, which is the task of identifying groups of trajectories that exhibit similar movement patterns. Despite multiple definitions of trajectory similarity, most prior work assumes that trajectories are similar if spatially close [171], making clustering based on movement dynamics rather than spatial proximity far less explored in the literature.

In this work, we instead shift the focus to evaluating these novel vectorization techniques on the task of clustering trajectories based on similar events, namely their movement dynamics, while disregarding the spatial region. Under this perspective, two trajectories are assigned to the same cluster if they exhibit similar movement patterns. For instance, two trajectories that include a harsh stop should be grouped into the same cluster, even if they occur in different geographical regions.

A further motivation for this chapter is the big-data perspective of the thesis. Raw trajectories are complex sequential objects and are often difficult to process with mature large-scale analytics tools. By transforming trajectories into vector-based representations, one can leverage a much broader ecosystem of off-the-shelf machine learning, clustering, indexing, and distributed processing methods that are already well developed for vector data. In this sense, trajectory vectorization is not only a representation choice, but also an enabling step for scalable mobility analytics, since there are far fewer mature big-data counterparts designed directly for raw mobility objects and trajectory-native operations.

The contribution of this work is fourfold:

- we propose a unified benchmarking framework for evaluating trajectory vectorization techniques in unsupervised settings;

- we introduce a movement-oriented dictionary-based discretization scheme that captures kinematic and geometric patterns independently of geographic location;
- we extend GEOLET with an unsupervised shapelet selection mechanism;
- through extensive experiments we demonstrate that different vectorization techniques encode complementary mobility characteristics.

The remainder of the work is organized as follows. Section 8.2 introduces the trajectory clustering problem and notation. Section 8.3 presents the proposed evaluation framework. Sections 8.4 and 8.5 describe the experimental setup and results. Sections 8.6 and 8.7 concludes and outlines future work.

8.2 Problem Formulation

This section introduces the problem formulation for trajectory clustering in unsupervised settings. It describes the input-output data according to the problem formulation and clarifies the role of trajectory vectorization in the clustering process.

Using a trajectory dataset X, the trajectory clustering problem addressed in this chapter is formulated as follows:

- Given:
  - (i) a trajectory dataset Xcomposed of n trajectories,
  - (ii) a vectorization function g, which maps each trajectory into a fixed-size feature representation T, and
  - (iii) a clustering function h, which maps the vectorized trajectories into cluster identities.
- Predict: the cluster identity $c \in C$ of each trajectory X, where clustering is defined as the composition of the vectorization and clustering functions, that is,

$$c = (h \circ g)(X).$$

The objective is to examine how different trajectory vectorization functions affect clustering quality, clustering diversity, and scalability in unsupervised settings, especially when similarity is defined in a dataset-dependent and potentially behavioral manner rather than only through spatial proximity.

Formally, we begin by defining the underlying trajectory data.

Definition (Trajectory).

A trajectory X is a sequence of spatio-temporal points

$$X = \{(lat_0, long_0, t_0), \dots, (lat_m, long_m, t_m)\} \in \mathbb{R}^{m \times 3},$$

where the spatial vectors $\vec{x}_j = \left(lat_j, long_j\right)$ are sorted by increasing timestamp t_j, that is, for all $j \in [1, m)$, we have

$$t_j < t_{j+1}.$$

This type of data is typically not sampled at a constant rate, since observations may be recorded irregularly due to sensor limitations, energy-saving policies, communication constraints, or event-driven acquisition

processes. In addition, trajectories within a dataset may contain different numbers of observations. Therefore, we denote by

$$X \in \mathbb{R}^{n \times m \times 3}$$

a dataset of n trajectories, each of length at most m.

Definition (Trajectory Clustering).

Given a trajectory dataset X, trajectory clustering is the task of assigning each trajectory to a cluster identity $c \in C$. In this chapter, clustering is defined as the composition of a vectorization function g, which maps each trajectory to a fixed-size feature vector T, and a clustering function h, which maps the vectorized representation $g(X)$ to a cluster label. Thus, for a trajectory X, the output is

$$c = (h \circ g)(X).$$

By varying the vectorization function g, this formulation accommodates different clustering objectives and different notions of trajectory similarity.

In contrast to classical trajectory clustering settings, where similarity is often interpreted mainly as spatial proximity between trajectories [171], the goal of this chapter is to study a more abstract and dataset-dependent notion of similarity. For example, given a dataset in which certain trajectories exhibit behavioral patterns such as harsh braking, abrupt turns, or sustained high-speed intervals, we examine whether different vectorizations can reveal these distinctions without using labeled data.

8.3 Methodology

In this section, we propose a unified benchmarking framework, summarized in Figure 20, to evaluate off-the-shelf trajectory vectorization techniques. Rather than asking whether two trajectories are geographically close, our framework allows us to explore how different vectorization techniques naturally group trajectories that share behavioral or structural similarities. To make this evaluation transparent and reproducible, we define a common pipeline and measure both (i) clustering quality within a representation space and (ii) diversity/consistency across representation spaces. The framework is composed of 3 modules: (i) a vectorization function g; (ii) an optional dimensionality reduction step; (iii) a clustering function h. In addition to clustering quality, the framework also includes a distributed clustering setting, allowing us to examine how vectorized trajectory representations can be processed with scalable analytics tools in a PySpark environment. Finally, the resulting trajectory clustering is evaluated both in terms of clustering quality within each vectorization and in terms of diversity/consistency across vectorization spaces.

In the following paragraphs, we motivate the inclusion of the three modules and outline the algorithms considered for each in the experimental section. The framework is general and can accommodate a wide range of algorithms per module; the choices made here are preliminary and will be extended in future work.

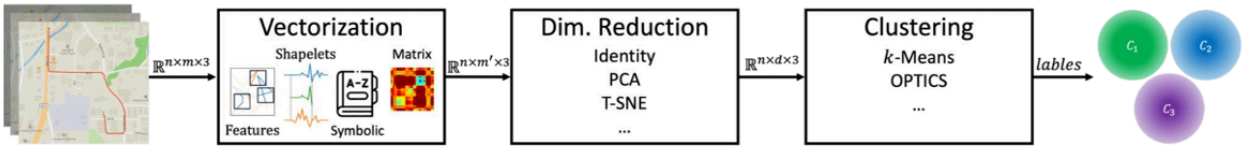


Figure 20. Overview of the proposed framework

Vectorization functions. The first step in our framework is selecting a set of vectorization functions to compare. The scope of this module is to transform complex trajectory data into a tabular-like format such that a classical off-the-shelf machine learning technique can be applied. Motivated by recent work in mobility, we select 4 different functions. Additionally, inspired by the time series literature, we divided them into feature-based, shapelet-based, dictionary-based, and matrix-based functions. For feature-based vectorization, i.e., where explicit descriptors are extracted from the data, we selected the Trajectory Interval Forest (TIF) [156]. TIF randomly generates a set of intervals and, for each interval and trajectory in the dataset, it extracts a wide range of descriptors (speed, acceleration, sinuosity, intensity use etc.). For shapelet-based vec- torization, which extracts a set of discriminatory subsequences and uses them as references for data transformation, we select GEOLET. GEOLET relies on statistical methods to identify discriminatory subsequences, and represents each trajectory by its similarity to the selected patterns. Since GEOLET was originally designed for supervised settings, its subsequence selection requires prior knowledge. In Section 8.3.1, we extend GEOLET to unsupervised tasks. Finding a dictionary-based vectorization was more challenging. In time series analysis, this family of methods discretizes the input into sequences of symbols, which are then matched against a learned dictionary of patterns. This allows temporal dynamics to be represented through the frequency or arrangement of symbolic motifs. In the mobility literature, discretization is typically performed using either tessellation techniques or pathlet-learning approaches [146], but these methods mainly capture global spatial characteristics of movement.

As a result, two harsh right turns occurring in different geographic regions would be encoded as different symbols. In Section 8.3.2, we propose a simple yet effective approach to address this limitation. Finally, for matrix-based vectorization, we selected the algorithm proposed in [153], namely RoSITa. This vectorization technique represents each trajectory in a scale- and rotation-invariant representation by performing the Hough transform.

Optional feature reduction. As some vectorization functions may produce very high-dimensional or sparse feature spaces, this step aims to mitigate the curse of dimensionality. Due to their widespread use among practitioners, we selected Principal Component Analysis (PCA) and t-distributed Stochastic Neighbor Embedding (t-SNE). We also introduced a pass-through option that allows the clustering algorithm to be executed directly on the full vectors.

Clustering algorithms. The third module applies an off-the-shelf clustering algorithm to the resulting (reduced) vectorized dataset. We consider two widely used methods from different families: *k*- means for centroid-based clustering and optics for density-based clustering. This selection will be expanded in future work.

8.3.1 GEOLET for Unsupervised Tasks

This section extends GEOLET to unsupervised tasks. GEOLET is composed of four steps: (i) trajectory segmentation into a set of candidate discriminative sub-trajectories, (ii) sub-trajectory normalization, (iii) sub-trajectory filtering to select the most discriminative candidates, and (iv) transformation of the dataset into a

dissimilarity matrix between each trajectory and the selected discriminative candidates. Since GEOLET was originally proposed for supervised tasks, the filtering step relies on dataset labels to identify, through statistical tests, the most relevant discriminative sub-trajectories. We therefore focus on replacing this step with a more general approach suitable for unsupervised settings. Inspired by time series literature, we build upon the concept of unsupervised shapelets (u-shapelets) [170], which extends the original notion of time series shapelets to scenarios where class labels are unavailable. The underlying intuition of the proposed work is that, even in unlabeled data, certain short subsequences tend to recur within groups of instances while remaining dissimilar to the rest of the dataset. Accordingly, a u-shapelet can be identified by assessing how effectively a candidate subsequence separates the dataset into two partitions: trajectories that contain a close match to the subsequence and those that do not.

We incorporate the following iterative procedure into GEOLET. Candidate subsequences are first extracted from a reference trajectory using spatial, temporal, or spatio-temporal sliding windows. For each candidate, a distance measure is used to compute a dissimilarity vector for all trajectories in the dataset. The distance measure is defined as the minimal distance over all possible alignments of the subsequence within each trajectory. Sorting this vector produces an orderline, over which all possible split points are evaluated. For each split, a gap score measures the separation between the two induced groups by balancing the difference in their mean distances against their respective variances. Given a subsequence S that partitions the dataset into two trajectory groups A and B, the gap score is formally defined as:

$$gap = \mu B - \sigma B - (\mu A + \sigma A) \tag{23}$$

Where μA and μB are the means of the best fitting of S in all trajectories in A and in B respectively, while σA and σB represent the standard deviations. The subsequence that maximizes this separation is selected as a discriminative subsequence. Trajectories that are well explained by the selected u-shapelet are then removed from the dataset. The process is repeated on the remaining instances to extract additional u-shapelets, until the desired number of u- shapelets has been extracted or no further splits of the remaining instances are possible.

8.3.2. Dictionary-based vectorization

This section lays the groundwork for shifting from continuous to discrete representations of trajectories. We introduce a set of algorithms designed to transform trajectory data into sequences of symbols. This discretization provides several important benefits: it enables the use of highly efficient data structures; it simplifies storage and indexing by reducing continuous data into compact symbolic forms; and it facilitates faster similarity searches and pattern discovery. We formally define the problem of trajectory discretization as the composition of a normalization function and a discretization function, i.e., discretize ◦ normalize.

Let a trajectory be represented as a sequence of spatio-temporal points in $\mathrm{R}^{m' \times 3}$, where m is the original number of observations. The normalization function $\mathrm{R}^{m \times 3} \rightarrow \mathrm{R}^{m' \times 3}$ transforms the original trajectory into a normalized version containing m' points. Here, m' denotes the number of observations after normalization. This step serves two purposes: first, it acts as an approximation function similar to PAA in SAX [159], reducing noise and redundancy; second, it regularizes the trajectory by interpolating or resampling to ensure a consistent sampling rate or spatial distance. This normalization ensures that trajectories of comparable "length" yield the same number of symbols, enabling more meaningful comparisons: trajectories exhibiting similar motion patterns are more likely to share a similar symbolic representation. The discretization function then maps the normalized trajectory to a symbolic sequence: $\mathrm{R}^{m' \times 3} \rightarrow \Sigma^{m'}$ where Σ is a finite alphabet of symbols and $\Sigma^{m'}$ denotes a sequence of m' symbols drawn from that alphabet. In other words, each normalized observation is assigned to one symbol, and the full normalized trajectory is transformed into a symbolic sequence of length m'.

This normalization and discretization process ensures that trajectories of comparable effective length yield symbolic sequences of comparable size, enabling more meaningful comparisons. As a result, trajectories exhibiting similar motion patterns are more likely to produce similar symbolic representations. Finally, the discretized trajectories can be vectorized by computing a TF-IDF matrix over symbolic "words" extracted through a sliding-window procedure.

Normalization

We present three possible normalization strategies suited for trajectory data. The first involves resampling trajectories at constant time intervals, ensuring that observations are uniformly spaced in time. The second normalizes by constant travel distance, where a new observation is sampled each time the moving object covers a fixed spatial distance. However, due to positional noise, this method may produce redundant samples when the object is stationary. To address this, we introduce a third variant that generates a new observation only after the object has moved a minimum distance from the last sampled point, rather than after a fixed cumulative distance. In all three strategies, missing observations are interpolated, and excess ones are subsampled to preserve temporal or spatial regularity.

Discretization

We present several strategies for assigning a sequence of symbols to a trajectory. A first, straightforward approach relies on tessellation techniques already established in the literature.

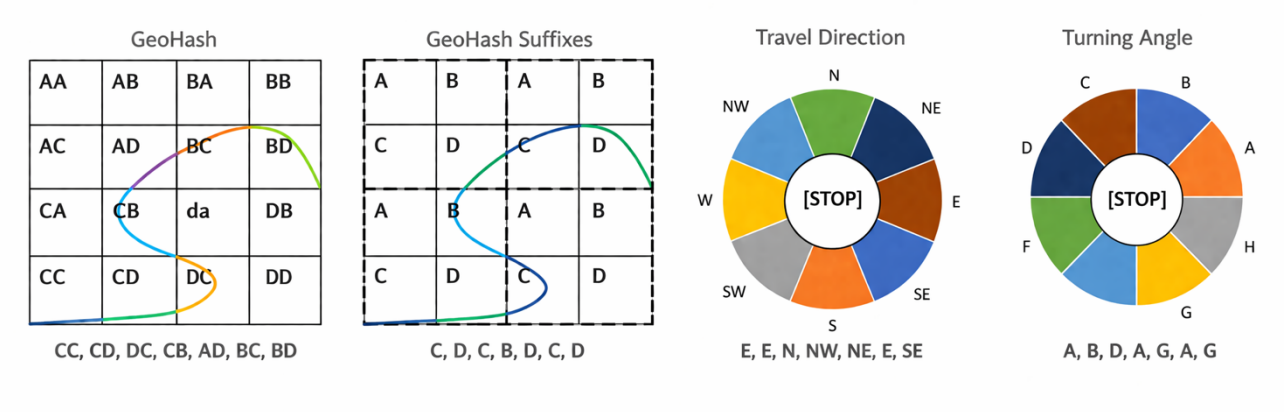


Figure 21. Strategies to transform trajectories into symbolic sequences.

Figure 21 (left) illustrates a conceptual example using Geohash [164], where each geohash region crossed by the trajectory generates a corresponding symbol1. The same idea can be extended to other spatial partitioning methods. However, since our goal is to analyze multiple aspects of trajectories beyond mere origin–destination movement, we aim to design a discretization strategy that yields comparable symbolic vectorizations across different geographic contexts. One possible approach is to leverage the hierarchical structure of some tessellation encodings, such as Geohash. Leveraging this property, we can design discretization techniques based on suffix relationships among GeoHash sequences, enabling vectorizations that remain structurally consistent even when trajectories span different geographic areas. Figure 21 (center-left) illustrates such an example. The main limitation of this approach is its reliance on a fixed spatial grid: two similar movements are represented by similar subsequences only if they occur within the same subregion. For instance, consider two right-turn events–one from cell "CB" to "AD" and another from "BC" to "BD". Despite being similar, the discretized subsequences differ ("CB" vs. "CD") solely due to their geographic position.

This spatial dependence limits the generalization of symbolic vectorizations across regions. To overcome this, we introduce two alternative discretization strategies that are independent of geographic location and instead rely on the intrinsic dynamics of motion. Specifically, the traveling direction and the turning angle. The first computes the direction between consecutive observations and assigns symbols based on predefined angular intervals, as shown in Figure 21 (center-right). The second focuses on the turning angle, i.e., the change in direction between successive segments. Revisiting the right-turn example, this vectorization consistently captures both events as the same symbolic pattern, generating the symbol “G” twice. The resulting longest common subsequence, “AG”, effectively captures a straight movement followed by a right turn, offering a location- and orientation-independent yet behaviorally meaningful vectorization. Additionally, to account for stationary behavior, we introduce a dedicated symbol that indicates when the object is not moving.

8.3.3 Distributed Clustering Methods

To scale the proposed vectorization-and-clustering framework to large trajectory collections, we implemented the clustering stage in a distributed environment using PySpark. The clustering stage is studied under two different execution settings. In the centralized setting, clustering is performed with standard single-machine implementations, where we consider methods such as K-Means and OPTICS in order to evaluate clustering quality under a fixed and controlled pipeline. In the distributed setting, the goal is different: rather than broadening the family of clustering algorithms, we examine whether vectorized trajectory data can be processed efficiently with scalable big-data tooling. For this purpose, we implement distributed K-Means in PySpark, using Apache Spark MLlib as a representative large-scale vector clustering environment.

This distinction is important because the centralized and distributed settings serve different purposes. The centralized setting supports methodological comparison among clustering approaches, whereas the distributed setting is used to assess the scalability potential of the vectorization-and-clustering pipeline once trajectories have been transformed into vector data. After preprocessing and vectorizing trajectories (e.g., tif, GEOLET-derived features, RoSITa, or the proposed symbolic tf–idf representation), each trajectory is represented as a fixed-length feature vector and stored as a Spark DataFrame. This enables the use of off-the-shelf clustering algorithms available in Spark MLlib, allowing model fitting to be parallelized across executors and applied efficiently to high-dimensional representations. In practice, the distributed pipeline supports standard clustering routines (e.g., partition-based methods such as k-means and Gaussian mixture models, where applicable), while preserving a uniform interface across different representations. This design keeps the methodology modular: representation learning is decoupled from clustering, and alternative algorithms or hyperparameter search strategies can be plugged in without modifying the upstream feature construction. Overall, the PySpark implementation provides a scalable and reproducible way to benchmark clustering behavior across multiple representation spaces and datasets, reducing runtime and enabling experiments that would be impractical in a single-machine setting.

8.4 Experimental Setup

This section describes the hyperparameters for vectorization, feature reduction, and clustering evaluated in our experiments. Due to limited prior knowledge of the effectiveness of these vectorization techniques for unsupervised tasks, we designed the experiments to evaluate all possible combinations of the three components. Consequently, each vectorization hyperparameter setting was evaluated with every feature reduction hyperparameter configuration, and each of those combinations was in turn evaluated with every clustering hyperparameter configuration. As the total number of experiments increases exponentially with the number of hyperparameters across modules, we prioritize extensive exploration of the vectorization hyperparameters while restricting the feature reduction and clustering stages to a smaller set of representative methods and settings.

8.4.1 Hyperparameters

Vectorization method hyperparameters

For TIF, we explored intervals defined by the number of observations, time, or distance-based intervals. The number of intervals ranged from 2 to 20, with different constraints on minimum and maximum lengths. We disabled the use of latitude and longitude to avoid making position-based tasks trivial. For GEOLET, we tested different numbers of subsequences, from 5 up to 100, using a Geohash-based partitioning with precision levels between 3 and 7. To select the most discriminative shapelets, we experimented with the algorithm described in the previous section. For the distance computation between trajectories and shapelets, we tested standard Euclidean, time-interpolated Euclidean, and space-interpolated Euclidean distances. For RoSITa, we used a time-based sliding window ranging from 10 seconds to 1 hour. For the Hough Transform, the angle resolution varied between 1° and 10°, and the image size ranged from 50×50 to 1080×1080 pixels. Consequently, the Hough accumulator had dimensions between approximately 70 × 18 and 1527 × 180. We tested clustering using the full accumulator, its vertically summed squared version, and its FFT transform as in [165]. For the dictionary-based approach, we evaluated all discretization and resampling strategies. Time-based resampling was tested every 1, 50, or 300 seconds, and distance-based resampling was tested every 3, 50, or 300 meters. For tessellation-based discretization, we used Geohash with precision levels 3 to 7 and suffix variants, ignoring the first one to three characters. Direction-based discretization employed alphabets of 4, 8, or 16 symbols, plus one for stationarity. We then built a tf-idf matrix over words extracted with sliding windows of 3 to 20 symbols.

Feature reduction hyperparameters

Regarding the feature reduction step, we evaluated several approaches commonly adopted in machine learning. In particular, we experimented with Principal Component Analysis (PCA) and t-distributed Stochastic Neighbor Embedding (t-SNE). For both methods, we tested output dimensionalities of 5, 10, 50, and 100 components. For t-SNE, we additionally varied the perplexity parameter over 5-100. All remaining hyperparameters for both PCA and t-SNE were left unchanged and set to their default values as defined in the scikit-learn implementation.

Clustering methods hyperparameters.

Finally, regarding the clustering step, we limited our experiments to *k*-means and OPTICS. For *k*-means, we evaluated values of *k* equal to 2, 3, 4, 6, and 10. For OPTICS, we tested two values for the minimum number of samples required in the neighborhood of a point for it to be considered a core point: 5 and 10% of the trajectories in the dataset. All remaining hyperparameters for both clustering methods were left at their default values3. Please note that clustering quality metrics were computed on the non-reduced vectorized representations. This choice was made to enable a fair comparison of clustering scores across different vectorization methods sharing the same hyperparameter configurations. Otherwise, such metrics would also be conditioned by the output of the feature reduction step, thereby confounding the evaluation of the vectorization methods themselves.

8.4.2 Datasets

The broader dataset perspective of the thesis was summarized in Section 2.3.1. In the present chapter, we focus specifically on the datasets selected for the benchmark of trajectory vectorization and clustering in

unsupervised settings, emphasizing the properties that are most relevant to representation quality, clustering behavior, and cross-dataset comparison.

We consider both real-world and synthetic datasets. This distinction allows us to capture different levels of similarity to real data. Synthetic datasets are fully controlled and used to test specific, isolated aspects of the vectorization functions.

Real-world datasets. We selected five real datasets 4 for GPS trajectory classification with different sizes, semantics, and classification objectives, i.e., animals, vehicles, seabirds, geolife. Table 19 (top five rows) reports descriptive statistics for the datasets, including the number of trajectories, the total number of observations, the average and standard deviation of observations per trajectory, the average sampling rate, and the number and proportion of target classes. For the animals dataset, the classification task is to recognize three species. For vehicles, the objective is to distinguish between buses and trucks. For seabirds, the objective is to recognize the flying trajectories of three species of seabirds. For the geolife dataset, we set the classification problem to recognize trajectories of public versus private means of transport as in [157].

Synthetic Datasets. For synthetic datasets, we use an extended version of those introduced in [155], consisting of a larger number of trajectories per dataset. The synthetic datasets are deliberately simple to isolate the hypothesis under analysis, minimizing confounding variables such as traffic, traffic lights, road priorities, etc. Additionally, trajectories in the synthetic datasets are spatially dense, i.e., similar movements occur in the same region.

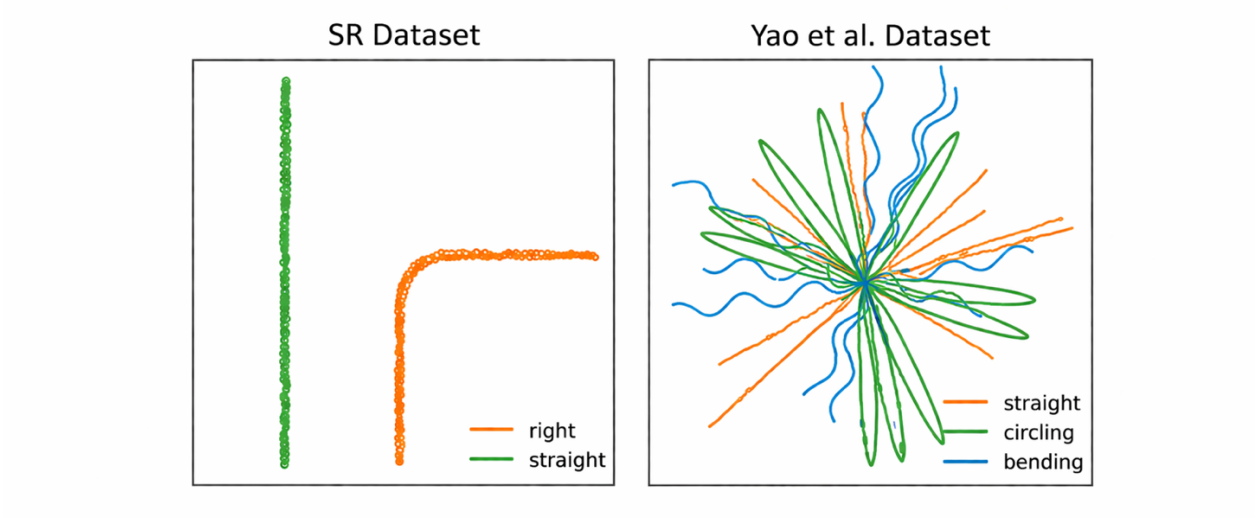


Figure 22. A sample of the two Shape-based datasets: Straight-Right dataset and Yao dataset.

To measure the extent to which different vectorization techniques capture the movement's geometry, we generated two synthetic datasets shown in Figure 22. The first dataset, Straight-Right (SR), consists of objects that either move in a straight line or make a 90-degree right turn. Each object in this dataset travels at a constant speed for precisely 100 seconds. The second dataset, Yao, is inspired by the dataset introduced by Yao et al. in [167]. It includes three distinct movement patterns: straight, circling, and bending. Unlike the SR dataset, this dataset introduces additional complexity to the analysis task, as each object travels in a different direction with a variable sampling rate for a varying number of seconds between 70 and 130. We generated 100 trajectories for each class in the dataset, resulting in 200 trajectories for the SR dataset and 300 for the Yao dataset.

		#trj	#obs	#obs per trj	Avg Δtime (s)	Classes
Real - World	Animals	102	15k	145 ± 62	4.85 ± 4.85	Cattle 33% Deer 29% Elk 37%
	Seabirds	108	264k	2.4 ± 1k	100 ± 0.0	tCOGU 29% tEUSH 14%
	Vehicles	381	178k	467 ± 251	122 ± 160	tRAZO 57% Bus 28%
	Geolife	5977	5.3M	892 ± 2.5k	265 ± 2.9k	Truck 72% Public T. 43% Private T. 57%
Synthetic	SR	200	20k	100 ± 0.0	1 ± 0.0	Straight 50% Turn 50%
	FS	200	20k	100 ± 0.0	3.0 ± 2.0	Slow 50% Fast 50%
	Yao	300	21k	69 ± 12	1.5 ± 0.7	Straight 33% Circling 33% Bending 33%

Table 19. Datasets metadata

Kinematic-related synthetic datasets. We generated a simple synthetic dataset –Fast Slow (FS)– of moving objects turning 90 degrees right and traveling at two different speeds. By isolating a specific structural movement pattern while varying the travel speed, we aim to assess the effectiveness of vectorization techniques in capturing movement dynamics without confounding effects from other variables. The target labels associated with the trajectories could be either moving “slow” or “fast”.

8.5 Results

In this section, we analyze how distinct vectorization captures structural patterns in the data under an unsupervised setting. Standard clustering practice often relies on the knee method or selects hyperparameters that maximize the silhouette (SIL) score. However, both heuristics depend on pairwise distances; hence, they are affected by the shape of the clusters. Additionally, clustering is applied in the vectorized space; therefore, these metrics can be used only when comparing clustering outcomes derived from the same vectoriza- tion type and configuration. In our framework, we compute the SIL score in the non-reduced vectorization spaces, enabling the use of this metric to compare clustering results across various feature reduction techniques. Additionally, since our datasets include class labels, we use them as a proxy for ground truth and select, for each component of the pipeline, the hyperparameters that maximize the Fowlkes–Mallows (FM) score. Alternative heuristics to select the best clustering results without any prior knowledge are left as future work. SIL and FM scores alone do not characterize differences between feature spaces; they only quantify cluster separation and agreement with the labeled structure. To evaluate how the relative positions of instances vary across vectorizations of different dimensionality, we exploit the Triplet Loss introduced in [163], a widely used one-shot learning loss function that encourages embeddings in which similar instances are close together, and dissimilar instances are far apart. In our framework, we adopt a variant called Random Triplet Accuracy (rta), proposed in [166], to measure the impact of feature reduction techniques. This measure operates by randomly sampling triplets of instances from two feature spaces and evaluating the proportion of triplets in which the relative distance ordering between instances is preserved across both spaces. A score near 1 indicates that both

spaces encode instance relationships consistently, while a score near 0 suggests that they reverse these relationships, placing samples that are close in one space farther apart in the other.

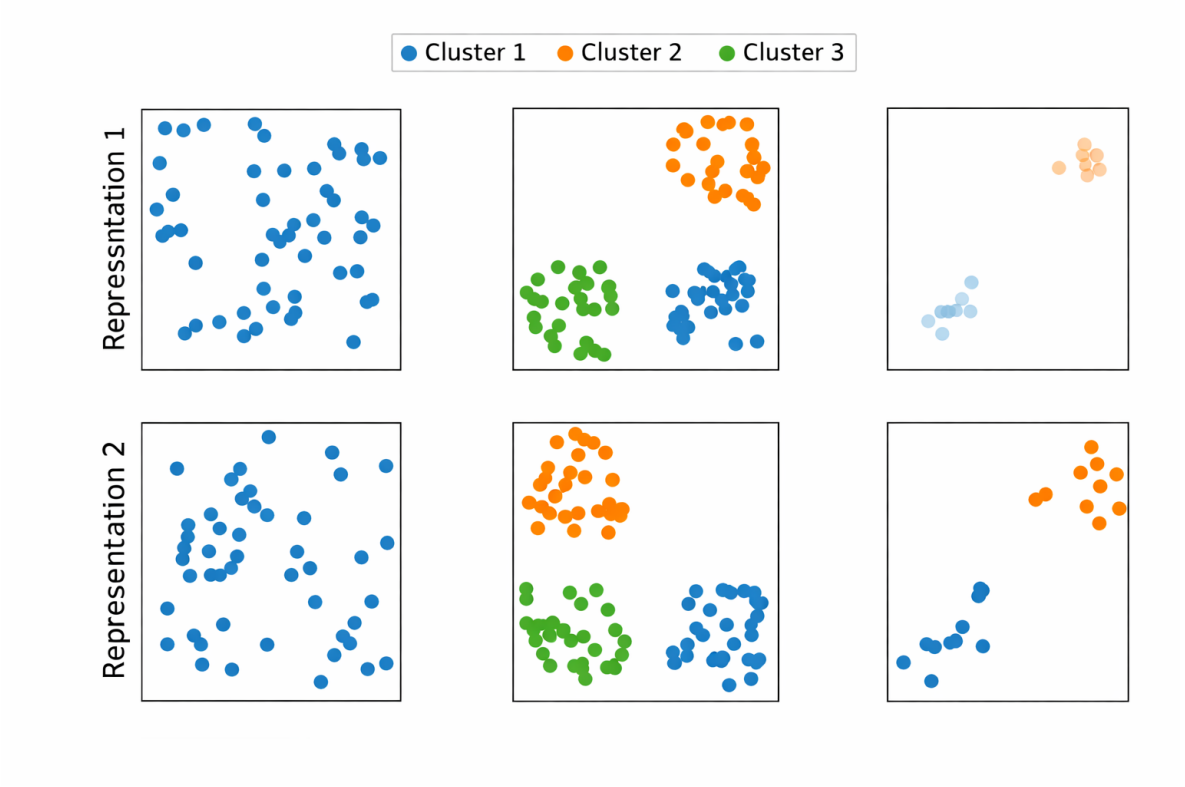


Figure 23. Comparison of RTA across three settings

Figure 23 illustrates three scenarios. In the first, 90 instances are randomly projected into two unrelated feature spaces. Because rta samples triplets at random, we repeat the evaluation 100 times and obtain an average score of ≈ 0.50, indicating that an ordering such as $\text{ed}(a, b) \leq \text{ed}(a, c)$ in one space fails to hold in the other in roughly half of the cases. In the second scenario, both vectorizations yield the same three clusters; however, the points are randomly arranged within each cluster, and the orange cluster is closer to different neighbors across the two spaces. Under the same experimental conditions, the resulting score is approximately 0.70. Note that such a scenario would have the same FM score and approximately the same SIL score. Finally, in the third scenario, the vectorizations differ only by a scale change, leading to an rta score close to 1.

	FM				SIL			
	TIF	GEOLET	RoSITA	TF-IDF	TIF	GEOLET	RoSITA	TF-IDF
				Best on FM				
Geometry	0.881	0.880	**0.993**		0.350	0.639	0.777	
Kinematic	**0.977**	0.709	0.890		**0.805**	0.659	-0.005	
Real-world	0.754	**0.781**	0.675	0.744	-0.004	0.437	**0.480**	0.429
All	**0.877**	0.790	0.852	0.744	0.384	**0.578**	0.417	0.429
				Best on SIL				
Geometry	**0.881**	**0.881**	**0.881**		0.824	0.867	**0.888**	
Kinematic	**0.993**	0.702	0.502		**0.945**	0.778	0.072	
Real-world	**0.661**	0.603	0.644	0.605	0.932	0.963	**0.975**	0.927
All	**0.847**	0.729	0.675	0.605	0.867	0.869	0.645	**0.927**

Table 20. Summary of experimental results on clustering quality.

Tables 20 and 21 present the experimental results on vectorization diversity, reported through the rta score, and clustering quality, reported through the SIL and FM scores. The optimal pipeline for each vectorization and dataset is selected using the FM score. Table 20 additionally reports the SIL score associated with the pipeline that maximizes the FM score, as well as the FM and SIL scores of the pipeline that performs best with respect to the SIL score, thereby simulating a selection process that relies solely on internal clustering criteria.

We begin by examining the upper part of Table 20, which reports the FM and SIL scores of the pipeline achieving the highest FM. To facilitate interpretation, FM and SIL scores are averaged across datasets with the same characteristics: shape-based, kinematic- based, and real-world trajectories. By selecting the pipeline that maximizes FM, tif achieves the overall best performance across all datasets. Notably, RoSITa performs particularly well on geometry- based datasets, which aligns with its design. Moreover, GEOLET consistently attains the overall best SIL scores, indicating an average higher degree of cluster separation w.r.t. other vectorization techniques, which can be attributed to its sub-trajectory filtering strategy based on the gap score. For the other vectorizations, we do not observe any particularly interesting pattern5. Instead, selecting the best pipeline that maximizes SIL yields slightly lower FM scores across all vectorization methods and dataset categories. On the other side, SIL scores are extremely high for virtually all datasets and vectorization techniques. Our interpretation of these results goes as follows: we found pipelines with high value for FM or SIL score, which means that, for some vectorization hyperparameters, each vectorization is able to discover clusters similar to the ground truth, or to produce vectorization spaces with well-separated clusters, but not as high similarity with the ground truth. While this highlights the flexibility of such vectorization in capturing higher level mobility patterns beyond mere spatial closeness, it also makes it difficult for practitioners in real-world scenarios to determine the best hyperparameters without additional information about the data.

Despite that, several consistent trends emerge from the results in Table 20. The tif vectorization demonstrates the highest overall FM values across almost all dataset categories. This indicates that the handcrafted feature extraction pipeline effectively captures both spatial and temporal properties of movement, producing clusters that closely correspond to the actual labeled classes. Conversely, GEOLET achieves superior performance on real-world datasets, where data are more complex, irregular, and noisy. This suggests

that shapelet-based vectorizations are especially well-suited to capture localized movement variations or subpatterns (e.g., turns, stops, or loops) that often define behavior in natural or human mobility. The relatively lower SIL values, combined with high FM scores, indicate that, while GEOLET clusters may be less compact geometrically, they still align closely with the semantic ground truth. The RoSITa vectorization performs best on geometry-related datasets, as expected due to its design for rotation- and scale-invariant shape encoding. This confirms that geometric alignment is particularly important when trajectories exhibit clear structural regularities. However, RoSITa's performance drops slightly on datasets dominated by dynamic differences, suggesting its invariance properties may suppress meaningful kinematic variation. Finally, the TF–IDF (dictionary-based) approach achieves competitive results on real world datasets, suggesting significant potential for this vectorization technique.

		TIF	GEOLET	RoSITA	TF-IDF
Kinematic	**TIF**	1.000 ± 0.000	0.540 ± 0.027	0.523 ± 0.014	-
	GEOLET	0.540 ± 0.027	1.000 ± 0.000	0.505 ± 0.011	-
	RoSITA	0.523 ± 0.014	0.505 ± 0.011	1.000 ± 0.000	-
	TF-IDF	-	-	-	-
Geometry	**TIF**	1.000 ± 0.000	0.601 ± 0.103	0.580 ± 0.110	-
	GEOLET	0.601 ± 0.103	1.000 ± 0.000	0.571 ± 0.120	-
	RoSITA	0.580 ± 0.110	0.571 ± 0.120	1.000 ± 0.000	-
	TF-IDF	-	-	-	-
Geom-Kin	**TIF**	1.000 ± 0.000	0.579 ± 0.051	0.609 ± 0.080	-
	GEOLET	0.579 ± 0.051	1.000 ± 0.000	0.600 ± 0.091	-
	RoSITA	0.609 ± 0.080	0.600 ± 0.091	1.000 ± 0.000	-
	TF-IDF	-	-	-	-
Real-world	**TIF**	1.000 ± 0.000	0.608 ± 0.044	0.548 ± 0.012	0.583 ± 0.049
	GEOLET	0.608 ± 0.044	1.000 ± 0.000	0.509 ± 0.017	0.499 ± 0.034
	RoSITA	0.548 ± 0.012	0.509 ± 0.017	1.000 ± 0.000	0.592 ± 0.035
	TF-IDF	0.583 ± 0.049	0.499 ± 0.034	0.592 ± 0.035	1.000 ± 0.000

Table 21. Summary of experimental results on clustering diversity using RTA.

Table 21 reports the rta scores between pairs of vectorizations across all dataset categories. Vectorization hyperparameters follow the configuration that yielded the highest FM score. Overall, the scores obtained by comparing different transformation spaces average around 0.543, with values reaching up to 0.714 for specific dataset–vectorization combinations. As illustrated in Figure 23, entirely unrelated vectorizations typically yield a score of about 0.500, while points that fall into the same clusters but differ in their cluster-wise positioning tend to produce scores near 0.700. Interpreting these results benefits from revisiting Table 20, particularly the FM values. Since the FM score, defined as the geometric mean of precision and recall, ranges from 0 to 1 and reflects cluster similarity, its generally high values here suggest that most rta scores correspond to cases where clusters are similar but differ in their internal positioning.

The highest rta values (≈ 0.714) occur between GEOLET and RoSITa for geometry-based datasets, suggesting that although the methods rely on different principles, they capture similar global structure. In contrast, the lowest rta values (≈ 0.500) appear between GEOLET and tf–idf in real-world datasets, reflecting a fundamental difference between local shapelet patterns and symbolic discretization. These relationships highlight the complementarity among vectorization families. From a practical perspective, the reported rta scores suggest promising research avenues for ensemble or hybrid vectorizations.



Figure 24. Two dimensional PCA projections of the representation spaces produced by TIF, GEOLET and RoSITA across SR, FS datasets. Colors indicate ground-truth clusters.

To qualitatively observe such complementarity, we now examine representative embedding spaces for the selected datasets. Figure 24 presents 2D PCA projections for tif, GEOLET, and RoSITa (left to right). For the SR dataset (top row), all three embedding spaces clearly separate instances by their true clusters. For FS (bottom row), where classes differ only in movement speed, tif yields a clean separation, whereas the 2D projection of the RoSITa space collapses the instances into a single point. Nonetheless, its FM score is nearly optimal, indicating that the collapse is a visual artefact of PCA. The GEOLET projection also appears scattered almost at random. However, its FM score of 0.705 suggests that, despite the imperfect visualization, meaningful separation persists in this balanced binary setting.

Finally, to examine the scalability perspective of the framework, we compared a distributed K-Means implementation using Apache Spark MLlib against the baseline scikit-learn KMeans under aligned clustering settings, including the same feature representation, number of clusters, initialization strategy, and stopping criteria. The experiments were executed in a single-node environment (MacBook Pro with Apple M1 Pro), where Spark exploits parallelism across local cores but also incurs JVM and orchestration overhead. Under this setting, Spark MLlib achieved a modest end-to-end runtime improvement of approximately 5–15% relative to scikit-learn.

These results should be interpreted carefully. They do not constitute full evidence of large-scale distributed superiority, since the experiments were not conducted on a multi-node cluster and the datasets, while nontrivial, were still within a regime where single-node execution remains feasible. Nevertheless, the experiment provides proof of concept that once trajectories are converted into vector representations, the resulting pipeline can be coupled with mature distributed analytics frameworks. In this sense, the main value of the experiment is not the absolute speedup itself, but the demonstration that trajectory vectorization opens the door to scalable off-the-shelf big-data processing.

8.6 Discussion

This chapter examined trajectory vectorization as a central methodological issue in unsupervised mobility analysis. By comparing representative vectorization families across synthetic and real-world datasets, the chapter showed that the choice of representation strongly influences clustering quality, cluster diversity, and the type of structure that can be recovered from movement data. In this sense, the results confirm that trajectory clustering cannot be understood independently of representation design.

A first important observation is that no single vectorization family dominates under all conditions. Methods that perform well on synthetic datasets with controlled structure do not necessarily transfer equally well to noisier real-world datasets. This is an important result because it shows that representation quality is not absolute; it depends on the type of trajectory variation that is most relevant in a given dataset, such as geometry, local subtrajectory structure, motion dynamics, or behavioral regularity.

A second observation is that different vectorizations preserve different aspects of movement. Feature-based approaches are effective when clustering depends strongly on motion dynamics and interpretable descriptive statistics. Shapelet-based or subtrajectory-based approaches are better suited to capturing localized geometric or behavioral motifs. Invariant representations such as RoSITa are particularly strong when geometric shape is the dominant clustering signal. These findings indicate that the vectorizations are not merely alternative encodings of the same information, but genuinely different views of the data.

A third observation is that clustering evaluation itself must be multidimensional. The chapter shows that relying on a single metric can lead to misleading conclusions. Internal criteria such as Silhouette can favor compact partitions that do not necessarily align with meaningful external structure, while external criteria such as FM capture agreement with known labels but may not reflect broader clustering geometry. The use of RTA adds a complementary perspective by revealing whether different representations preserve similar or distinct structural views of the dataset. Taken together, these metrics support a more robust benchmarking methodology.

A fourth observation concerns the broader role of vectorization in the thesis. Earlier chapters focused mainly on forecasting, where representation was important but often remained embedded inside the prediction pipeline. In the present chapter, representation becomes the primary object of study. This is significant because it highlights that mobility analytics is not only about designing stronger predictors, but also about deciding how raw movement data should be encoded before any downstream task is applied.

A fifth observation relates directly to the big-data perspective of the dissertation. One of the most important implications of vectorization is that it enables trajectory data to be processed with mature off-the-shelf machine-learning and distributed analytics tools. This is particularly relevant in large-scale settings, where raw sequential mobility objects are often difficult to analyze directly with scalable general-purpose systems. The distributed K-Means experiment in this chapter should therefore be interpreted as proof of concept rather than as conclusive evidence of large-scale distributed superiority. Its main value lies in showing that once trajectories are converted into vector representations, they can be coupled naturally with existing big-data infrastructures.

At the same time, the chapter also reveals the current limits of this scalability claim. The observed runtime gains in the Spark-based experiment are modest, and the evaluation was conducted in a single-node environment rather than a multi-node distributed setting. Thus, the chapter does not claim that vectorization alone solves the big-data problem of trajectory analytics. Rather, it shows that vectorization creates the necessary compatibility with scalable tools, thereby opening a realistic path toward larger-scale mobility analysis.

Within the broader structure of the thesis, Chapter 8 is important because it completes the methodological arc of the dissertation from a different direction than the forecasting chapters. Chapters 3–6 studied forecasting at different abstraction levels and under increasing scalability demands. Chapter 7 examined internal neural-model design through trainable activation functions. Chapter 8 turns to the representation layer itself and shows that the encoding of trajectories is a central factor affecting both clustering behavior and the possibility of scalable downstream analysis.

Overall, the results support a broader thesis-level conclusion: in mobility analytics, representation is not merely a preprocessing step but a core methodological choice. It shapes the structure that can be discovered, the robustness of clustering outcomes, and the compatibility of trajectory data with general-purpose learning and large-scale processing frameworks.

8.7 Summary

This chapter introduced a benchmark framework for trajectory vectorization in unsupervised settings and used it to compare alternative representation families across multiple datasets. The results showed that vectorization choice has a substantial effect on clustering quality, clustering diversity, and the type of mobility structure that can be recovered from the data. In this sense, the chapter established that unsupervised trajectory analysis depends as much on representation design as on the clustering algorithm itself.

The experiments also showed that different vectorization families preserve complementary information. Feature-based methods, shapelet-based approaches, dictionary-based representations, and invariant encodings each emphasize different aspects of movement behavior and geometry. As a result, no single method should be expected to dominate universally across all datasets and clustering objectives. This finding justifies the need for a benchmarking framework that evaluates representations systematically rather than through isolated case studies.

An additional contribution of the chapter is methodological. By combining internal and external clustering criteria with representation-diversity analysis, the framework provides a richer and more reliable basis for evaluating trajectory vectorizations. This strengthens the broader thesis emphasis on controlled evaluation and explicit methodological comparison.

Beyond clustering quality alone, the chapter also highlighted an important systems implication of trajectory vectorization. By mapping raw trajectories into fixed-size vector spaces, it becomes possible to apply mature off-the-shelf machine-learning, clustering, indexing, and distributed processing tools that are much harder to use directly on raw sequential mobility objects. In this way, the chapter connects representation design to the big-data perspective of the dissertation and shows that vectorization can serve as an enabling step for scalable mobility analytics.

Within the overall structure of the thesis, Chapter 8 completes the representation-oriented part of the dissertation. After the earlier chapters examined forecasting paradigms, unified architectures, scalable forecasting, and neural-model methodology, the present chapter showed that the way mobility data are represented is itself a central research problem. This closes the main methodological arc of the thesis by reinforcing the broader view that data mining in big mobility data is simultaneously a problem of prediction, representation, evaluation, and systems design.

The material presented in this chapter is based on and extends the publication “A Benchmark Framework for Trajectory Vectorization in Unsupervised Settings” [154].

Part VI Consolidation and Outlook

Chapter 9: Conclusion

This thesis addressed the broader problem of data mining in big mobility data through a coherent sequence of contributions spanning trajectory representation, individual trajectory forecasting, aggregate traffic flow forecasting, unified and distributed forecasting architectures, and methodological advances in deep learning. Across these contributions, the central objective was not only to improve predictive accuracy, but also to clarify how mobility problems should be formulated, how alternative paradigms should be compared, how learning pipelines can remain feasible under large-scale conditions, and how modular methodological improvements can extend beyond a single model or task.

A first major contribution of the thesis concerned individual mobility forecasting, studied through Vessel Route Forecasting (VRF). Chapter 3 established a controlled experimental benchmark for route forecasting using statistical, machine learning, and neural approaches under realistic AIS-derived conditions. The key conclusion from this part of the thesis is that trajectory forecasting quality depends not only on the predictive model itself, but also on how one handles irregularity, sparsity, trajectory segmentation, and evaluation design. In this sense, Chapter 3 did more than compare algorithms: it clarified what it means to benchmark route forecasting rigorously in big mobility data settings.

The thesis then moved from individual movement to aggregate mobility forecasting, studied through Vessel Traffic Flow Forecasting (VTFF). Chapter 4 showed that aggregate forecasting is not a single modeling problem but a family of formulations with important methodological trade-offs. In particular, it clarified the distinction between direct approaches, which predict future flow values directly from past aggregate flow observations, and indirect approaches, which first predict individual movement and then aggregate it into future flows. This comparison established an important thesis-wide insight: the level of abstraction at which mobility is modeled strongly affects what information is preserved, what computational cost is incurred, and what types of operational support the resulting model can provide.

Building on this insight, Chapter 5 introduced UA-VTFF, a unified framework that combines vessel-level and flow-level predictive information. The key conclusion from this chapter is that hybridization across forecasting paradigms can improve robustness and predictive behavior, especially under sparse or dynamically changing traffic conditions. Rather than treating direct and indirect approaches as competing alternatives, UA-VTFF showed that they can be interpreted as complementary views of the same mobility process. From a broader perspective, this contribution illustrates a recurrent message of the thesis: in mobility analytics, useful prediction often emerges when different abstractions of the data are combined rather than isolated.

Chapter 6 extended this unified perspective toward distributed and scalable forecasting through dUA-VTFF. This contribution addressed a central reality of big mobility data: accurate models are insufficient if the surrounding data-processing and learning pipeline cannot operate feasibly at realistic scales. By integrating distributed processing with a Transformer-based forecasting design, Chapter 6 demonstrated that scalability must be treated as a scientific requirement and not merely as an implementation detail. The contribution of this chapter therefore lies both in predictive performance and in the explicit treatment of computational feasibility as part of the forecasting problem itself.

The thesis then turned to a complementary methodological question in Chapter 7: how can one improve neural forecasting models without redesigning their overall architecture? The answer explored here was through KANB, a family of trainable parametric activation functions based on cubic Bézier curves. This chapter contributed to the thesis by showing that activation design is not a minor hyperparameter choice, but a methodological degree of freedom that can materially influence convergence behavior and predictive performance. Importantly, KANB was evaluated not only in forecasting settings but also in image-classification experiments, which supports the conclusion that the contribution is broader than a single

maritime forecasting architecture. In this way, Chapter 7 reinforced the thesis objective of contributing not only applications, but also reusable deep-learning methodology.

Finally, Chapter 8 addressed trajectory representation and clustering in unsupervised settings, providing the broadest and most domain-general perspective of the thesis. This chapter showed that representation is not merely a preprocessing step, but a foundational modeling decision that affects clustering quality, similarity structure, and the usability of trajectories in downstream analytics. By benchmarking alternative vectorization families across datasets and by examining both clustering quality and computational considerations, the chapter clarified how representation choices preserve or distort different aspects of mobility behavior. It also strengthened the thesis' broader perspective beyond maritime forecasting by showing that trajectory analytics decisions can generalize across mobility domains.

Taken together, the contributions of Chapters 3–8 form an integrated research pipeline. The thesis begins with how mobility should be predicted at the object level, moves to how it should be forecast at the aggregate level, then to how these views can be unified, then to how such forecasting can remain feasible at scale, then to how the neural building blocks themselves can be improved, and finally to how mobility should be represented for unsupervised analysis and scalable downstream learning. This progression is not accidental: it reflects the thesis' underlying view that mobility analytics is simultaneously a problem of representation, prediction, systems design, and methodological robustness.

Several cross-cutting conclusions emerge from this overall progression. First, problem formulation matters. Route forecasting, aggregate flow forecasting, and unsupervised trajectory clustering are not interchangeable tasks, and their value depends on what operational question is being asked. Second, hybridization is repeatedly useful. The thesis demonstrated this at several levels: combining direct and indirect forecasting, combining distributed systems with learning models, and combining adaptive neural components with existing architectures. Third, scalability is inseparable from relevance in big mobility data. A method cannot be considered practically meaningful if it cannot be trained, evaluated, or deployed under realistic data volumes and update conditions. Fourth, representation remains central even when prediction is the main target, since what is encoded, preserved, and compared shapes all later modeling steps.

The practical implications of these conclusions extend beyond the immediate case studies. In maritime operations, the forecasting contributions support collision-risk awareness, regional traffic supervision, port and route planning, and potentially emissions-aware operational strategies. More broadly, the methodological lessons regarding abstraction, unification, distributed learning, and representation are relevant to other mobility domains such as aviation and urban transportation, where noisy trajectories, large data volumes, and the need for proactive decision support are equally central. The broader framing established in Chapters 1, 2, 7, and 8 suggests that the thesis should be read not only as a collection of maritime forecasting studies, but as a structured contribution to mobility data mining more generally.

At the same time, it is important to state clearly what the thesis does not claim. Most empirical forecasting results are centered on maritime AIS data, and while Chapter 8 and part of Chapter 7 broaden the scope across domains and tasks, broader claims of universal generality would require further cross-domain experimentation. Similarly, the distributed and scalable contributions provide strong evidence of feasibility and methodological direction, but they do not eliminate all open questions concerning streaming deployment, concept drift, uncertainty handling, and operational integration. These limitations do not weaken the thesis contribution; rather, they define its scope clearly and motivate the next set of research directions.

Overall,this thesis contributes:

- rigorous and comparable evidence for individual route forecasting,
- a structured characterization of aggregate flow forecasting paradigms,

- unified and distributed forecasting frameworks that connect object-level and flow-level prediction while addressing scalability,
- a transferable methodological deep-learning contribution through trainable activation functions, and
- a benchmark framework for trajectory vectorization in unsupervised settings that broadens the thesis toward general mobility representation and clustering.

In this sense, the thesis establishes a coherent and extensible foundation for future work on accurate, scalable, and methodologically robust mobility analytics.

Chapter 10: Future Work

The contributions of this thesis establish a strong foundation for mobility analytics, but they also open several important directions for future research. These directions arise directly from the structure of the dissertation itself: from route forecasting to aggregate flow forecasting, from unified and distributed learning to methodological deep-learning improvements, and from trajectory representation to unsupervised analytics. More broadly, the future-work agenda should not be framed only as a continuation of the empirical case studies presented here, but as an extension of the thesis pipeline toward more adaptive, uncertainty-aware, scalable, explainable, and cross-domain mobility intelligence systems.

A first major direction is multimodal context integration. Most forecasting components in the thesis rely primarily on movement traces and derived spatiotemporal features. However, many forecasting failures arise not from insufficient model capacity but from missing information about the operating context. Future work should therefore integrate heterogeneous data sources such as weather and sea-state conditions, bathymetry and coastline constraints, port schedules, traffic regulations, radar feeds, satellite imagery, and other domain-specific metadata. In maritime forecasting, such information could improve robustness under unusual operating conditions such as storms, route diversions, port disruptions, or temporary exclusion zones. More generally, multimodal fusion would also support broader transferability, since similar combinations of movement and contextual data exist in aviation and urban mobility.

A second important direction is uncertainty-aware forecasting. The present thesis emphasizes predictive accuracy and comparative robustness, but operational mobility systems often require more than point predictions. Future forecasting frameworks should estimate not only future positions or flow values, but also the uncertainty associated with those predictions. This includes probabilistic route forecasting, prediction intervals for traffic flow, and calibrated confidence measures for decision support. Such extensions are especially important in safety-critical domains, where the value of a forecast depends not only on its mean accuracy but also on how reliably the system communicates what it does and does not know.

A third direction concerns real-time and streaming deployment. The thesis has already advanced the scalability perspective through distributed processing and learning, particularly in Chapter 6. A natural next step is to move from distributed batch-style workflows toward genuinely streaming mobility analytics. This would require architectures capable of low-latency data ingestion, incremental feature updates, online recalibration, and possibly continual or adaptive learning. In practical terms, future work could extend dUA-VTFF into a streaming system that combines event ingestion and distributed processing with real-time inference and alerting. Scientifically, this introduces further questions about concept drift, stability of online updates, and how forecasting models should be evaluated under continuously evolving data streams.

A fourth direction is cross-domain validation and transferability. One of the broader goals of the thesis has been to frame mobility analytics generically rather than as a collection of maritime-only results. Future work should therefore examine explicitly which thesis components transfer to aviation and urban mobility, and under what modifications. For route forecasting, this means studying whether object-level methods developed for AIS can generalize to ADS-B or GPS settings with different update regimes and movement constraints. For aggregate flow forecasting, this means comparing grid-based and graph-based formulations across maritime, urban, and aviation settings. For representation learning, it means extending the benchmark of Chapter 8 to more real-world datasets and more diverse unsupervised tasks. Such work would move the thesis from conceptual genericity to broader empirical validation.

A fifth direction concerns the extension of unified forecasting architectures. Chapters 5 and 6 showed that direct and indirect forecasting can be integrated in useful ways, and that such integration can be scaled computationally. Future research should explore richer forms of unification, including architectures that

combine multiple spatial resolutions, adaptive fusion mechanisms that weight route-level and flow-level information dynamically, and models that incorporate heterogeneous graph or network structure instead of relying only on regular grid-based discretization. These extensions would make the unified perspective more flexible and potentially more robust across different operating conditions and mobility domains.

A sixth direction concerns explainability and human-centered mobility analytics. As forecasting models become more complex and are increasingly deployed in operational settings, understanding their decisions becomes more important. Future work should therefore examine how route forecasts, flow predictions, clustering results, and activation-function behavior can be interpreted in ways that are useful to domain experts. In mobility analytics, explainability is not only a technical property but also an operational requirement, since the usefulness of a model often depends on whether decision-makers can understand the reasoning behind its outputs.

A seventh direction concerns representation learning beyond clustering. Chapter 8 showed that trajectory vectorization is central to unsupervised analysis and to the compatibility of mobility data with scalable general-purpose learning tools. Future work could extend this direction toward self-supervised representation learning, contrastive trajectory modeling, multi-vectorization fusion, and task-adaptive representations that support both prediction and downstream unsupervised tasks. Such research would strengthen the bridge between forecasting and representation, which is one of the broader themes of the dissertation.

An eighth direction concerns evaluation methodology itself. One of the recurring themes of the thesis is that performance claims in mobility analytics depend strongly on preprocessing choices, target definitions, abstraction levels, and evaluation design. Future work should therefore continue to strengthen the benchmarking philosophy developed in this dissertation by incorporating more datasets, broader baselines, stronger ablation studies, more explicit computational reporting, and more realistic operational evaluation scenarios. In this sense, benchmarking remains not only a support activity, but an important research direction in its own right.

Taken together, these future directions suggest that mobility analytics will continue to evolve along several interacting dimensions: richer data integration, more expressive but reliable forecasting models, scalable and streaming learning pipelines, stronger representation learning, and broader cross-domain validation. The contributions of this thesis provide a foundation for these directions by showing that mobility data mining should be treated simultaneously as a problem of prediction, representation, evaluation, and systems design.

The broader outlook of the dissertation is therefore one of methodological integration. Future progress in mobility analytics is unlikely to come from one isolated innovation alone. It will more likely emerge from the interaction of better forecasting formulations, stronger representations, scalable infrastructures, uncertainty-aware learning, and explainable decision support. In this sense, the thesis is intended not only as a collection of chapter-level contributions, but as a structured step toward a more mature and general framework for intelligent mobility data mining.

REFERENCES

[1] A. Artikis and D. Zissis, Guide to Maritime Informatics. Springer, 2021.

[2] H. Georgiou, P. Petrou, P. Tampakis, S. Sideridis, E. Chondrodima, N. Pelekis, and Y. Theodoridis, "Future location and trajectory prediction," in Big Data Analytics for Time-Critical Mobility Forecasting: From Raw Data to Trajectory-Oriented Mobility Analytics in the Aviation and Maritime Domains. Cham: Springer, 2020, pp. 215–254.

[3] P. Tampakis, E. Chondrodima, A. Pikrakis, Y. Theodoridis, K. Pristouris, H. Nakos, E. Petra, T. Dalamagas, A. Kandiros, G. Markakis, I. Maina, and S. Kavadas, "Sea area monitoring and analysis of fishing vessels activity: The i4sea big data platform," in Proceedings of the IEEE International Conference on Mobile Data Management (MDM), 2020, pp. 275–280.

[4] P. Tampakis, E. Chondrodima, A. Tritsarolis, A. Pikrakis, Y. Theodoridis, K. Pristouris, H. Nakos, P. Kalampokis, and T. Dalamagas, "i4sea: A big data platform for sea area monitoring and analysis of fishing vessels activity," Geo-spatial Information Science, 2021, doi: 10.1080/10095020.2021.1971055.

[5] E. Tu, G. Zhang, L. Rachmawati, E. Rajabally, and G.-B. Huang, "Exploiting ais data for intelligent maritime navigation: A comprehensive survey from data to methodology," IEEE Transactions on Intelligent Transportation Systems, vol. 19, no. 5, pp. 1559–1582, 2018.

[6] D. Zissis, E. K. Xidias, and D. Lekkas, "Real-time vessel behavior prediction," Evolving Systems, vol. 7, pp. 29–40, 2016.

[7] N. Zorbas, D. Zissis, K. Tserpes, and D. Anagnostopoulos, "Predicting object trajectories from high-speed streaming data," in Proceedings of the IEEE International Conference on Trust, Security and Privacy in Computing and Communications / IEEE International Conference on Big Data Science and Engineering / IEEE International Conference on Embedded Software and Systems, vol. 2, 2015, pp. 229–234.

[8] A. Valsamis, K. Tserpes, D. Zissis, D. Anagnostopoulos, and T. Varvarigou, "Employing traditional machine learning algorithms for big data streams analysis: The case of object trajectory prediction," Journal of Systems and Software, vol. 127, pp. 249 – 257, 2017.

[9] S. Wang, J. Cao, and P. Yu, "Deep learning for spatio-temporal data mining: A survey," IEEE Transactions on Knowledge and Data Engineering, 2020.

[10] S. Capobianco, L. M. Millefiori, N. Forti, P. Braca, and P. Willett, "Deep learning methods for vessel trajectory prediction based on recurrent neural networks," IEEE Transactions on Aerospace and Electronic Systems, vol. 57, no. 6, pp. 4329–4346, 2021, doi: 10.1109/TAES.2021.3096873.

[11] Y. Suo, W. Chen, C. Claramunt, and S. Yang, "A ship trajectory prediction framework based on a recurrent neural network," Sensors, vol. 20, no. 18, 2020.

[12] Y. Wang, D. Zhang, Y. Liu, and K. Tan, "Trajectory forecasting with neural networks: An empirical evaluation and a new hybrid model," IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 10, pp. 4400–4409, 2020.

[13] R. Trasarti, R. Guidotti, A. Monreale, and F. Giannotti, "Myway: Location prediction via mobility profiling," Information Systems, vol. 64, pp. 350– 367, 2017.

[14] P. Petrou, P. Nikitopoulos, P. Tampakis, A. Glenis, N. Koutroumanis, G. M. Santipantakis, K. Patroumpas, A. Vlachou, H. V. Georgiou, E. Chondrodima, C. Doulkeridis, N. Pelekis, G. L. Andrienko, F. Patterson, G. Fuchs, Y. Theodoridis, and G. A. Vouros, "ARGO: A big data framework for online trajectory prediction," in Proceedings of the International Symposium on Spatial and Temporal Databases (SSTD), 2019, pp. 194–197.

[15] H. Georgiou, N. Pelekis, S. Sideridis, D. Scarlatti, and Y. Theodoridis, “Semantic-aware aircraft trajectory prediction using flight plans,” International Journal of Data Science and Analytics, vol. 9, no. 2, pp. 215–228, 2020.

[16] P. B. Weerakody, K. W. Wong, G. Wang, and W. Ela, “A review of irregular time series data handling with gated recurrent neural networks,” Neurocomputing, vol. 441, pp. 161–178, 2021.

[17] D. Neil, M. Pfeiffer, and S.-C. Liu, “Phased LSTM: Accelerating recurrent network training for long or event-based sequences,” in Advances in Neural Information Processing Systems, vol. 29, 2016, pp. 3889–3897.

[18] Z. Xiao, X. Fu, L. Zhang, and R. S. M. Goh, “Traffic pattern mining and forecasting technologies in maritime traffic service networks: A comprehensive survey,” IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 5, pp. 1796–1825, 2020.

[19] N. Pelekis, P. Tampakis, M. Vodas, C. Panagiotakis, and Y. Theodoridis, “In-DBMS sampling-based sub-trajectory clustering,” in Proceedings of the International Conference on Extending Database Technology (EDBT), 2017, pp. 632–643.

[20] G. Pallotta, S. Horn, P. Braca, and K. Bryan, “Context-enhanced vessel prediction based on Ornstein–Uhlenbeck processes using historical AIS traffic patterns: Real-world experimental results,” in Proceedings of the International Conference on Information Fusion, 2014, pp. 1–7.

[21] G. Pallotta, M. Vespe, and K. Bryan, “Vessel pattern knowledge discovery from ais data: A framework for anomaly detection and route prediction,” Entropy, vol. 15, no. 6, pp. 2218–2245, 2013.

[22] D. Rumelhart, G. Hinton, and R. Williams, “Learning representations by back-propagating errors,” Nature, vol. 323, pp. 533–536, 1986.

[23] T. G. Dietterich, “Machine learning for sequential data: A review,” in Structural, Syntactic, and Statistical Pattern Recognition, T. Caelli, A. Amin, R. P. W. Duin, D. de Ridder, and M. Kamel, Eds. Berlin, Heidelberg: Springer, 2002, pp. 15–30.

[24] D. Montgomery and G. Runger, Applied Statistics and Probability for Engineers (7th/Ed.). John Wiley & Sons, 2018.

[25] J. Cervantes, F. Garcia-Lamont, L. Rodriguez-Mazahua, and A. Lopez, “A comprehensive survey on support vector machine classification: Applications, challenges and trends,” Neurocomputing, vol. 408, pp. 189–215, May 2020.

[26] V. Vapnik, The nature of statistical learning theory. Springer science & business media, 2013.

[27] C. Cortes and V. Vapnik, “Support-vector networks,” Machine learning, vol. 20, no. 3, pp. 273–297, 1995.

[28] S. Theodoridis and K. Koutroumbas, Pattern Recognition, 4th ed. Academic Press, November 2008.

[29] A. Ali, “Ensemble learning model for prediction of natural gas spot price based on least squares boosting algorithm,” in Proceedings of the International Conference on Data Analytics and Business Intelligence (ICDABI), 2020, pp. 1–6.

[30] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 785–794.

[31] T. K. Ho, “Random decision forests,” in Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), vol. 1, 1995, pp. 278–282.

[32] R. E. Schapire, “Explaining AdaBoost,” in Empirical Inference. Springer, 2013, pp. 37–52.

[33] S. Haykin, Neural Networks: A Comprehensive Foundation, 2nd ed. USA: Prentice Hall PTR, 1998.

[34] A. Poznyak, W. Yu, E. Sanchez, and J. Perez, “Nonlinear adaptive trajectory tracking using dynamic neural networks,” IEEE Transactions on Neural Networks, vol. 10, no. 6, pp. 1402–1411, 1999.

[35] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Computing, vol. 9, no. 8, pp. 1735–1780, 1997.

[36] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder–decoder for statistical machine translation,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1724–1734.

[37] P. J. Werbos, “Backpropagation through time: what it does and how to do it,” Proceedings of the IEEE, vol. 78, no. 10, pp. 1550–1560, 1990.

[38] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proceedings of the International Conference on Learning Representations (ICLR), 2015.

[39] M. Veres and M. Moussa, “Deep learning for intelligent transportation systems: A survey of emerging trends,” IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 8, pp. 3152–3168, 2019.

[40] S. Malakar, S. Goswami, B. Ganguli, A. Chakrabarti, S. S. Roy, K. Boopathi, and A. Rangaraj, “A novel feature representation for prediction of global horizontal irradiance using a bidirectional model,” Machine Learning and Knowledge Extraction, vol. 3, no. 4, pp. 946–965, 2021.

[41] K. Udeh, D. W. Wanik, N. Bassill, and E. Anagnostou, “Time series modeling of storm outages with weather mesonet data for emergency preparedness and response,” in Proceedings of the IEEE Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON), 2019, pp. 499–505.

[42] A. Graves and J. Schmidhuber, “Framewise phoneme classification with bidirectional lstm and other neural network architectures,” Neural Networks, vol. 18, no. 5, pp. 602 – 610, 2005.

[43] C. Ray, R. Dréo, E. Camossi, A.-L. Jousselme, and C. Iphar, “Heterogeneous integrated dataset for maritime intelligence, surveillance, and reconnaissance,” Data in Brief, vol. 25, p. 104141, 2019.

[44] G. Parsons, J. Youden, and C. Fowler, “Satellite automatic identification system (SAIS) performance modelling and simulation,” Defence R&D Canada–Ottawa, 2013.

[45] A. Alexandridis, E. Chondrodima, and H. Sarimveis, “Radial basis function network training using a nonsymmetric partition of the input space and particle swarm optimization,” IEEE Transactions on Neural Networks and Learning Systems, vol. 24, no. 2, pp. 219–230, 2013.

[46] L. Prechelt, “Early stopping — but when?” in Neural Networks: Tricks of the Trade. Berlin, Heidelberg: Springer, 1998, pp. 55–69.

[47] A. Alexandridis, E. Chondrodima, and H. Sarimveis, “Cooperative learning for radial basis function networks using particle swarm optimization,” Applied Software Computing, vol. 49, pp. 485 – 497, 2016.

[48] D.-W. Gao, Y.-S. Zhu, J.-F. Zhang, Y.-K. He, K. Yan, and B.-R. Yan, “A novel MP-LSTM method for ship trajectory prediction based on AIS data,” Ocean Engineering, vol. 228, p. 108956, 2021.

[49] B. Zhang, Z. Xu, J. Zhang, and G. Wu, “A warning framework for avoiding vessel-bridge and vessel-vessel collisions based on generative adversarial and dual-task networks,” Computer-Aided Civil and Infrastructure Engineering, 2021.

[50] C. Tang, J. Sun, Y. Sun, M. Peng, and N. Gan, “A general traffic flow prediction approach based on spatial-temporal graph attention,” IEEE Access, vol. 8, pp. 153731–153741, 2020.

[51] X. Wang, J. Li, and T. Zhang, “A machine-learning model for zonal ship flow prediction using AIS data: A case study in the South Atlantic States region,” Journal of Marine Science and Engineering, vol. 7, p. 463, 2019.

[52] P. Doganis, E. Aggelogiannaki, and H. Sarimveis, “A combined model predictive control and time series forecasting framework for productioninventory systems,” International Journal of Production Research., vol. 46, no. 24, pp. 6841–6853, 2008.

[53] W. He, C. Zhong, M. A. Sotelo, X. Chu, X. Liu, and Z. Li, “Short-term vessel traffic flow forecasting by using an improved Kalman model,” Cluster Computing, vol. 22, no. 4, pp. 7907–7916, 2019.

[54] X. Zhou, Z. Liu, F. Wang, Y. Xie, and X. Zhang, “Using deep learning to forecast maritime vessel flows,” Sensors, vol. 20, no. 6, p. 1761, 2020.

[55] D. Alizadeh, A. A. Alesheikh, and M. Sharif, “Prediction of vessels locations and maritime traffic using similarity measurement of trajectory,” Annals of GIS, pp. 1–12, 2020.

[56] D. Nguyen, C. L. Van, and M. I. Ali, “Vessel trajectory prediction using sequence-to-sequence models over spatial grid,” in Proceedings of the ACM International Conference on Distributed and Event-Based Systems (DEBS), 2018, pp. 258–261.

[57] E. Chondrodima, P. Mandalis, N. Pelekis, and Y. Theodoridis, “Machine learning models for vessel route forecasting: An experimental comparison,” in Proceedings of the IEEE International Conference on Mobile Data Management (MDM), 2022.

[58] N. Forti, L. M. Millefiori, P. Braca, and P. Willett, “Prediction oof vessel trajectories from ais data via sequence-to-sequence recurrent neural networks,” in Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2020, pp. 8936–8940.

[59] Z. Xiao, L. Ponnambalam, X. Fu, and W. Zhang, “Maritime traffic probabilistic forecasting based on vessels’ waterway patterns and motion behaviors,” IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 11, pp. 3122–3134, 2017.

[60] Z.-G. Zhang, J.-C. Yin, N.-N. Wang, and Z.-G. Hui, “Vessel traffic flow analysis and prediction by an improved PSO-BP mechanism based on AIS data,” Evolving Systems, vol. 10, no. 3, pp. 397–407, 2019.

[61] K.-I. Kim and K. M. Lee, “Deep learning-based caution area traffic prediction with automatic identification system sensor data,” Sensors, vol. 18, no. 9, p. 3172, 2018.

[62] H. Zhang, Y. Xiao, X. Bai, and L. Chen, “GA-support vector regression based ship traffic flow prediction,” International Journal of Control and Automation, vol. 9, no. 2, pp. 219–228, 2016.

[63] Z. Xie and Q. Liu, “LSTM networks for vessel traffic flow prediction in inland waterway,” in Proceedings of the IEEE International Conference on Big Data and Smart Computing (BigComp), 2018, pp. 418–425.

[64] S. Mouzakitis, C. Kontzinos, P. Kapsalis, I. Kanellou, G. Kormpakis, G. Tsapelas, D. Askounis, Optimising maritime processes via artificial intelligence: The vesselai concept and use cases in Proceedings of the International Conference on Information, Intelligence, Systems and Applications (IISA), 2022, pp. 1–5.

[65] Z. Xiao, X. Fu, L. Zhang,W. Zhang, R.W. Liu, Z. Liu, R. S. M. Goh, Big data driven vessel trajectory and navigating state prediction with adaptive learning, motion modeling and particle filtering techniques, IEEE Transactions on Intelligent Transportation Systems (2020).

[66] M. Papadimitrakis and A. Alexandridis, “Active vehicle suspension control using road preview model predictive control and radial basis function networks,” Applied Soft Computing, vol. 120, p. 108646, 2022.

[67] D. Karamichailidou, S. Koletsios, and A. Alexandridis, “An RBF online learning scheme for non-stationary environments based on fuzzy means and Givens rotations,” Neurocomputing, vol. 501, pp. 370–386, 2022.

[68] A. Nowy, K. Łazuga, L. Gucma, A. Androjna, M. Perkovič, and J. Srše, “Modeling of vessel traffic flow for waterway design—Port of Świnoujście case study,” Applied Sciences, vol. 11, p. 8126, 2021.

[69] P. Mandalis, E. Chondrodima, Y. Kontoulis, N. Pelekis, and Y. Theodoridis, “Machine learning models for vessel traffic flow forecasting: An experimental comparison,” in Proceedings of the IEEE International Conference on Mobile Data Management (MDM), 2022, pp. 431–436.

[70] M. Papadimitrakis, M. Stogiannos, H. Sarimveis, A. Alexandridis, “Multi-ship control and collision avoidance using MPC and RBF-based trajectory predictions,” Sensors, vol. 21, p. 6959, 2021.

[71] S. Mittal, S. Mahendra, V. Sanap, and P. Churi, “How can machine learning be used in stress management: A systematic literature review of applications in workplaces and education,” International Journal of Information Management Data Insights, vol. 2, p. 100110, 2022.

[72] A. C. Harvey, “ARIMA models,” in Forecasting, Structural Time Series Models and the Kalman Filter. London: Palgrave Macmillan, 1990, pp. 22–24.

[73] C. Tandon, S. Revankar, and S. S. Parihar, “How can we predict the impact of social media messages on the value of cryptocurrency? Insights from big data analytics,” International Journal of Information Management Data Insights, vol. 1, 2021.

[74] Y. Ensafi, S. H. Amin, G. Zhang, and B. Shah, “Time series forecasting of seasonal item sales using machine learning: A comparative analysis,” International Journal of Information Management Data Insights, vol. 2, p. 100058, 2022.

[75] E. Chondrodima, H. Georgiou, N. Pelekis, Y. Theodoridis, “Particle swarm optimization and RBF neural networks for public transport arrival time prediction using GTFS data,” International Journal of Information Management Data Insights, vol. 2, p. 100086, 2022.

[76] N. Korovesis, D. Kandris, G. Koulouras, and A. Alexandridis, “Robot motion control via an EEG-based brain-computer interface by using neural networks and alpha brainwaves,” Electronics, vol. 8, p. 1387, 2019.

[77] Y. Li and H. Ren, “Vessel traffic flow prediction using LSTM encoder-decoder,” in Proceedings of the International Conference on Signal Processing and Machine Learning (SPML), 2022, pp. 1–7..

[78] N. S. Gargari, R. Panahi, H. Akbari, and A. K. Ng, “Long-term traffic forecast using neural network and seasonal autoregressive integrated moving average: Case of a container port,” Transportation Research Record, vol. 2676, pp. 236–252, 2022.

[79] Z. Dong, “Prediction of ship traffic flow based on wavelet decomposition and LSTM,” in Proceedings of the International Conference on Cloud Computing, Big Data and Artificial Intelligence (ICCCBDA), 2022, pp. 88–93.

[80] D. Wang, Y. Meng, S. Chen, C. Xie, and Z. Liu, “A hybrid model for vessel traffic flow prediction based on wavelet and Prophet,” Journal of Marine Science and Engineering, vol. 9, 2021.

[81] H. Xiao, Y. Zhao, and H. Zhang, “Predict vessel traffic with weather conditions based on multimodal deep learning,” Journal of Marine Science and Engineering, vol. 11, p. 39, 2023.

[82] H. Rong, A. Teixeira, and C. G. Soares, “Maritime traffic probabilistic prediction based on ship motion pattern extraction,” Reliability Engineering & System Safety, vol. 217, p. 108094, 2022.

[83] T. Xu and Q. Zhang, “Ship traffic flow prediction in wind farms water area based on spatiotemporal dependence,” Journal of Marine Science and Engineering, vol. 10, p. 295, 2022.

[84] P. Mandalis, E. Chondrodima, Y. Kontoulis, N. Pelekis, and Y. Theodoridis, “A transformer-based method for vessel traffic flow forecasting,” GeoInformatica, vol. 29, no. 1, pp. 149–173, 2025, doi: 10.1007/s10707-024-00521-z.

[85] M. Stogiannos, M. Papadimitrakis, H. Sarimveis, and A. Alexandridis, “Vessel trajectory prediction using radial basis function neural networks,” in Proceedings of the IEEE EUROCON International Conference on Smart Technologies, 2021, doi: 10.1109/EUROCON52738.2021.9535562.

[86] M. Zhang, D. Zhang, S. Fu, P. Kujala, and S. Hirdaris, “A predictive analytics method for maritime traffic flow complexity estimation in inland waterways,” Reliability Engineering & System Safety, vol. 220, p. 108317, 2022.

[87] S. Dai, Y. Yu, H. Fan, and J. Dong, “Spatio-temporal representation learning with social tie for personalized POI recommendation,” Data Science and Engineering, vol. 7, pp. 44–56, 2022.

[88] H. Qi and B. Yin, “Deep learning on traffic prediction: Methods, analysis, and future directions,” IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 6, pp. 4927–4943, 2022, doi: 10.1109/TITS.2021.3054840.

[89] S. Guo, Y. Lin, S. Li, Z. Chen, and H. Wan, “Deep spatial-temporal 3D convolutional neural networks for traffic data forecasting,” IEEE Transactions on Intelligent Transportation Systems, vol. 20, no. 10, pp. 3913–3926, 2019, doi: 10.1109/TITS.2019.2906365.

[90] H. Li, H. Jiao, and Z. Yang, “AIS data-driven ship trajectory prediction modelling and analysis based on machine learning and deep learning methods,” Transportation Research Part E: Logistics and Transportation Review, vol. 175, p. 103152, 2023.

[91] F. Dai, P. Huang, Q. Mo, X. Xu, M. Bilal, and H. Song, “ST-INNet: Deep spatiotemporal inception networks for traffic flow prediction in smart cities,” IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 10, pp. 19782–19794, 2022, doi: 10.1109/TITS.2022.3179789.

[92] A. Alexandridis, H. Sarimveis, and K. Ninos, “A radial basis function network training algorithm using a non-symmetric partition of the input space—Application to a model predictive control configuration,” Advances in Engineering Software, vol. 42, no. 10, pp. 830–837, 2011, doi: 10.1016/j.advengsoft.2011.05.026.

[93] P. Mandalis, E. Chondrodima, Y. Kontoulis, N. Pelekis, and Y. Theodoridis, “Towards a unified vessel traffic flow forecasting framework,” in Proceedings of the Workshop on Big Mobility Data Analytics (BMDA), co-located with EDBT/ICDT, 2023.

[94] M. Gao and Y. Wei, “Traffic prediction based on a novel deep learning model: TPBERT,” in Proceedings of the International Conference on Artificial Intelligence and Big Data (ICAIBD), 2023, pp. 241–248, doi: 10.1109/ICAIBD57115.2023.10206156.

[95] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in Advances in Neural Information Processing Systems, vol. 30, 2017.

[96] P. Huang, Q. Chen, D. Wang, M. Wang, X. Wu, and X. Huang, “TripleConvTransformer: A deep learning vessel trajectory prediction method fusing discretized meteorological data,” Frontiers in Environmental Science, vol. 10, 2022.

[97] D. Jiang, G. Shi, N. Li, L. Ma, W. Li, and J. Shi, “TRFM-LS: Transformer-based deep learning method for vessel trajectory prediction,” Journal of Marine Science and Engineering, vol. 11, no. 4, 2023, doi: 10.3390/jmse11040880.

[98] M. Zaman, S. Saha, and S. Abdelwahed, “Assessing the suitability of different machine learning approaches for smart traffic mobility,” in Proceedings of the IEEE Transportation Electrification Conference & Expo (ITEC), 2023, pp. 1–6, doi: 10.1109/ITEC55900.2023.10186901.

[99] H. Yan, X. Ma, and Z. Pu, “Learning dynamic and hierarchical traffic spatiotemporal features with transformer,” IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 11, pp. 22386–22399, 2022, doi: 10.1109/TITS.2021.3102983.

[100] L. Zhu, F. R. Yu, Y. Wang, B. Ning, and T. Tang, “Big data analytics in intelligent transportation systems: A survey,” IEEE Transactions on Intelligent Transportation Systems, vol. 20, no. 1, pp. 383–398, 2019, doi: 10.1109/TITS.2018.2815678.

[101] A. Agafonov and A. Yumaganov, “Short-term traffic flow forecasting using a distributed spatial-temporal k-nearest neighbors model,” in Proceedings of the IEEE International Conference on Cyber Security and Cloud Computing / IEEE International Conference on Edge Computing and Scalable Cloud (CSE), 2018, pp. 91–98, doi: 10.1109/CSE.2018.00019.

[102] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, “Traffic flow prediction with big data: A deep learning approach,” IEEE Transactions on Intelligent Transportation Systems, vol. 16, no. 2, pp. 865–873, 2015, doi: 10.1109/TITS.2014.2345663.

[103] R. Fu, Z. Zhang, and L. Li, “Using LSTM and GRU neural network methods for traffic flow prediction,” in Proceedings of the Youth Academic Annual Conference of the Chinese Association of Automation (YAC), 2016, pp. 324–328, doi: 10.1109/YAC.2016.7804912.

[104] L. Cai, K. Janowicz, G. Mai, B. Yan, and R. Zhu, “Traffic transformer: Capturing the continuity and periodicity of time series for traffic forecasting,” Transactions in GIS, vol. 24, no. 3, pp. 736–755, 2020.

[105] Y. Fang, F. Zhao, Y. Qin, H. Luo, and C. Wang, “Learning all dynamics: Traffic forecasting via locality-aware spatio-temporal joint transformer,” IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 12, pp. 23433–23446, 2022, doi: 10.1109/TITS.2022.3197640.

[106] L. Huang, F. Zhu, and Z. Li, “TrendFormer: Trend adaptive transformer for traffic flow prediction,” in Proceedings of the International Conference on Data Science and Information Technology (DSIT), 2022, pp. 1–6, doi: 10.1109/DSIT55514.2022.9943858.

[107] E. Chondrodima, N. Pelekis, A. Pikrakis, and Y. Theodoridis, “An efficient LSTM neural network-based framework for vessel location forecasting,” IEEE Transactions on Intelligent Transportation Systems, vol. 24, no. 5, pp. 4872–4888, 2023, doi: 10.1109/TITS.2023.3247993.

[108] M. Usta, H. I. Turkmen, and M. A. Guvensan, “Distant and recent historical data fusion for improving short- and medium-term traffic forecasting,” Applied Sciences, vol. 15, no. 24, p. 13130, 2025.

[109] A. Troupiotis-Kapeliaris, D. Zissis, K. Bereta, M. Vodas, G. Spiliopoulos, and G. Karantaidis, “The big picture: An improved method for mapping shipping activities,” Remote Sensing, vol. 15, no. 21, p. 5080, 2023, doi: 10.3390/rs15215080.

[110] Apache Spark. Accessed on April 10, 2024. https://spark.apache.org/docs/2.1.3/ programming-guide.html.

[111] G. Kanakoudis, P. Mandalis and N. Pelekis, "KANB: A Family of Parametric Activation Functions With Trainable Parameters," in IEEE Access, vol. 13, pp. 146081-146087, 2025, doi: 10.1109/ACCESS.2025.3600484.

[112] E. Russo, C. Silvestri, M. Simeoni, A. Soares, and S. Matwin, “Predicting fishing effort and catch using semantic trajectories and machine learning,” in Multiple-Aspect Analysis of Semantic Trajectories: First International Workshop, MASTER 2019, Held in Conjunction with ECML-PKDD 2019, 2020, p. 83.

[113] J. Qu, R. W. Liu, Y. Guo, Y. Lu, J. Su, and P. Li, “Improving maritime traffic surveillance in inland waterways using the robust fusion of AIS and visual data,” Ocean Engineering, vol. 275, p. 114198, 2023.

[114] E. L. Manibardo, I. Laña, and J. Del Ser, “Deep learning for road traffic forecasting: Does it make a difference?” IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 7, pp. 6164–6188, 2022, doi: 10.1109/TITS.2021.3083957.

[115] S. Zhang, Y. Guo, P. Zhao, C. Zheng, and X. Chen, “A graph-based temporal attention framework for multi-sensor traffic flow forecasting,” IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 7, pp. 7743–7758, 2022, doi: 10.1109/TITS.2021.3072118.

[116] B. Varga, M. Pereira, B. Kulcsár, L. Pariota, and T. Péni, “Data-driven distance metrics for kriging-short-term urban traffic state prediction,” IEEE Transactions on Intelligent Transportation Systems, vol. 24, no. 6, pp. 6268–6279, 2023, doi: 10.1109/TITS.2023.3251022.

[117] H. Sun, “Traffic flow prediction model based on attention spatiotemporal graph convolutional network,” in Proceedings of the International Symposium on Computer Technology and Information Science (ISCTIS), 2023, pp. 148–153, doi: 10.1109/ISCTIS58954.2023.10213060.

[118] Y. Zhao, Y. Lin, H. Wen, T. Wei, X. Jin, and H. Wan, “Spatial-temporal position aware graph convolution networks for traffic flow forecasting,” IEEE Transactions on Intelligent Transportation Systems, vol. 24, no. 8, pp. 8650–8666, 2023, doi: 10.1109/TITS.2022.3220089.

[119] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” The Bulletin of Mathematical Biophysics, vol. 5, no. 4, pp. 115–133, 1943, doi: 10.1007/BF02478259.

[120] J. A. Akodia, C. K. Dzidonu, D. K. Boison, and P. Kisembe, “Fusion of activation functions: An alternative to improving prediction accuracy in artificial neural networks,” World Journal of Engineering and Technology, vol. 12, no. 4, pp. 836–850, 2024, doi: 10.4236/wjet.2024.124051.

[121] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, 2015, doi: 10.1038/nature14539.

[122] D. Mazzei and R. Ramjattan, “Machine learning for industry 4.0: A systematic review using deep learning-based topic modelling,” Sensors, vol. 22, no. 22, p. 8641, 2022, doi: 10.3390/s22228641.

[123] Y. Bodyanskiy and S. Kostiuk, “Adaptive hybrid activation function for deep neural networks,” System Research and Information Technologies, no. 1, pp. 87–96, 2022, doi: 10.20535/srit.2308-8893.2022.1.07.

[124] R. H. K. Emanuel, P. D. Docherty, H. Lunt, and K. Möller, “The effect of activation functions on accuracy, convergence speed, and misclassification confidence in CNN text classification: A comprehensive exploration,” The Journal of Supercomputing, vol. 80, no. 1, pp. 292–312, 2024, doi: 10.1007/s11227-023-05441-7.

[125] A. D. Jagtap and G. E. Karniadakis, “How important are activation functions in regression and classification? A survey, performance comparison, and future directions,” Journal of Machine Learning for Modeling and Computing, vol. 4, no. 1, pp. 21–75, 2023, doi: 10.1615/JMachLearnModelComput.2023047367.

[126] V. Kunc and J. Klema, “Three decades of activations: A comprehensive survey of 400 activation functions for neural networks,” arXiv:2402.09092, 2024.

[127] S. R. Dubey, S. K. Singh, and B. B. Chaudhuri, “Activation functions in deep learning: A comprehensive survey and benchmark,” Neurocomputing, vol. 503, pp. 92–108, 2022, doi: 10.1016/j.neucom.2022.06.111.

[128] H.-S. Feng and C.-H. Yang, “PolyLU: A simple and robust polynomial-based linear unit activation function for deep learning,” IEEE Access, vol. 11, pp. 101347–101358, 2023, doi: 10.1109/ACCESS.2023.3315308.

[129] X. Glorot, A. Bordes, and Y. Bengio, “Deep sparse rectifier neural networks,” in Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS), 2011, pp. 315–323.

[130] V. Nair and G. E. Hinton, “Rectified linear units improve restricted Boltzmann machines,” in Proceedings of the International Conference on Machine Learning (ICML), 2010, pp. 807–814.

[131] R. H. R. Hahnloser, R. Sarpeshkar, M. A. Mahowald, R. J. Douglas, and H. S. Seung, “Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit,” Nature, vol. 405, no. 6789, pp. 947–951, 2000.

[132] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, “Fast and accurate deep network learning by exponential linear units (ELUs),” in Proceedings of the International Conference on Learning Representations (ICLR), 2016.

[133] P. Ramachandran, B. Zoph, and Q. V. Le, “Searching for activation functions,” in Proceedings of the International Conference on Learning Representations (ICLR), Workshop Track, 2018.

[134] D. Misra, “Mish: A self regularized non-monotonic activation function,” arXiv:1908.08681, 2019.

[135] M. Asaduzzaman, M. Shahjahan, and K. Murase, “Faster training using fusion of activation functions for feed forward neural networks,” International Journal of Neural Systems, vol. 19, no. 6, pp. 437–448.

[136] J. K. Kruschke and J. R. Movellan, “Benefits of gain: Speeded learning and minimal hidden layers in back-propagation networks,” IEEE Transactions on Systems, Man, and Cybernetics, vol. 21, no. 1, pp. 273–280, 1991.

[137] Z. Hu and H. Shao, “The study of neural network adaptive control systems,” Control and Decision, vol. 7, pp. 361–366, 1992.

[138] C.-T. Chen and W.-D. Chang, “A feedforward neural network with function shape autotuning,” Neural Networks, vol. 9, no. 4, pp. 627–641, 1996, doi: 10.1016/0893-6080(96)00006-8.

[139] E. Trentin, “Networks with trainable amplitude of activation functions,” Neural Networks, vol. 14, nos. 4–5, pp. 471–493, 2001, doi: 10.1016/S0893-6080(01)00028-4.

[140] F. Agostinelli, M. Hoffman, P. Sadowski, and P. Baldi, “Learning activation functions to improve deep neural networks,” in Proceedings of the International Conference on Learning Representations (ICLR) Workshop, 2015.

[141] Y. Bodyanskiy and S. Kostiuk, “Learnable extended activation function for deep neural networks,” International Journal of Computing, pp. 311–318, 2023, doi: 10.47839/ijc.22.3.3225.

[142] J. A. Adams, “Computational geometry for design and manufacture,” Computer-Aided Design, vol. 11, no. 4, p. 243, 1979.

[143] S. A. Ahmed, D. P. Dogra, S. Kar, and P. P. Roy, “Trajectory-based surveillance analysis: A survey,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 29, no. 7, pp. 1985–1997, 2019.

[144] I. Ben-Gal, S. Weinstock, G. Singer, and N. Bambos, “Clustering users by their mobility behavioral patterns,” ACM Transactions on Knowledge Discovery from Data, vol. 13, no. 4, pp. 45:1–45:28, 2019.

[145] J. Bian, D. Tian, Y. Tang, and D. Tao, “A survey on trajectory clustering analysis,” arXiv:1802.06971, 2018.

[146] C. Chen, H. Su, Q.-X. Huang, L. Zhang, and L. J. Guibas, “Pathlet learning for compressing and planning trajectories,” in Proceedings of the ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, 2013, pp. 382–385.

[147] C. Leite da Silva, L. M. Petry, and V. Bogorny, “A survey and comparison of trajectory classification methods,” in Proceedings of the Brazilian Conference on Intelligent Systems (BRACIS), 2019, pp. 788–793.

[148] S. Dodge, R. Weibel, and E. Forootan, “Revealing the physics of movement: Comparing the similarity of movement characteristics of different types of moving objects,” Computers, Environment and Urban Systems, vol. 33, no. 6, pp. 419–434, 2009.

[149] M. F. Mokbel et al., “Towards mobility data science (vision paper),” arXiv:2307.05717, 2023.

[150] M. Etemad, A. Soares Júnior, and S. Matwin, “Predicting transportation modes of GPS trajectories using feature engineering and noise removal,” in Proceedings of the Canadian Conference on Artificial Intelligence, vol. 10832, 2018, pp. 259–264.

[151] C. A. Ferrero, L. O. Alvares, W. Zalewski, and V. Bogorny, “MOVELETS: Exploring relevant subtrajectories for robust trajectory classification,” in Proceedings of the ACM Symposium on Applied Computing (SAC), 2018, pp. 849–856.

[152] A. Hamdi, K. B. Shaban, A. Erradi, A. Mohamed, S. K. Rumi, and F. D. Salim, “Spatiotemporal data mining: A survey on challenges and open problems,” Artificial Intelligence Review, vol. 55, no. 2, pp. 1441–1488, 2022.

[153] C. Landi, N. Andrienko, and G. Andrienko, “Rotation- and scale-invariant shape extraction from vessel trajectories for human-in-the-loop monitoring,” in Proceedings of the ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, 2025.

[154] C. Landi, P. Mandalis, and N. Pelekis, “A benchmark framework for trajectory vectorization in unsupervised settings,” Workshop on Big Mobility Data Analytics (BMDA), 2026.

[155] C. Landi and R. Guidotti, “Shape-based methods in mobility data analysis: Effectiveness and limitations,” GeoInformatica, vol. 29, no. 4, pp. 999–1032, 2025.

[156] C. Landi, R. Guidotti, M. Nanni, and A. Monreale, “The trajectory interval forest classifier for trajectory classification,” in Proceedings of the ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, 2023, pp. 67:1–67:4.

[157] C. Landi, F. Spinnato, R. Guidotti, A. Monreale, and M. Nanni, “Geolet: An interpretable model for trajectory classification,” in Proceedings of the International Symposium on Intelligent Data Analysis (IDA), Lecture Notes in Computer Science, vol. 13876, 2023, pp. 236–248.

[158] J.-G. Lee, J. Han, X. Li, and H. Gonzalez, “TraClass: Trajectory classification using hierarchical region-based and trajectory-based clustering,” Proceedings of the VLDB Endowment, vol. 1, no. 1, pp. 1081–1094, 2008.

[159] J. Lin, E. J. Keogh, L. Wei, and S. Lonardi, “Experiencing SAX: A novel symbolic representation of time series,” Data Mining and Knowledge Discovery, vol. 15, no. 2, pp. 107–144, 2007.

[160] M. Middlehurst, A. Ismail-Fawaz, A. Guillaume, C. Holder, D. Guijo-Rubio, G. Bulatova, L. Tsaprounis, L. Mentel, M. Walter, P. Schäfer, and A. J. Bagnall, “aeon: A Python toolkit for learning from time series,” Journal of Machine Learning Research, vol. 25, no. 289, pp. 1–10, 2024.

[161] M. Musleh and M. F. Mokbel, “Let’s speak trajectories: A vision to use NLP models for trajectory analysis tasks,” ACM Transactions on Spatial Algorithms and Systems, 2024.

[162] C. Parent, S. Spaccapietra, C. Renso, G. Andrienko, N. Andrienko, V. Bogorny, M. L. Damiani, A. Gkoulalas-Divanis, J. A. F. de Macêdo, N. Pelekis, Y. Theodoridis, and Z. Wei Yan, “Semantic trajectories modeling and analysis,” ACM Computing Surveys, vol. 45, no. 4, pp. 42:1–42:32, 2013.

[163] F. Schroff, D. Kalenichenko, and J. Philbin, “FaceNet: A unified embedding for face recognition and clustering,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 815–823.

[164] I. S. Suwardi, D. Dharma, D. P. Satya, and D. P. Lestari, “Geohash index based spatial data model for corporate,” in Proceedings of the International Conference on Electrical Engineering and Informatics (ICEEI), 2015, pp. 478–483.

[165] M. Vlachos, Z. Vagena, P. S. Yu, and V. Athitsos, “Rotation invariant indexing of shapes and line drawings,” in Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM), 2005, pp. 131–138.

[166] Y. Wang, H. Huang, C. Rudin, and Y. Shaposhnik, “Understanding how dimension reduction tools work: An empirical approach to deciphering t-SNE, UMAP, TriMap, and PaCMAP for data visualization,” Journal of Machine Learning Research, vol. 22, pp. 201:1–201:73, 2021.

[167] D. Yao, C. Zhang, Z. Zhu, J.-H. Huang, and J. Bi, “Trajectory clustering via deep representation learning,” in Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN), 2017, pp. 3880–3887.

[168] L. Ye and E. J. Keogh, “Time series shapelets: A novel technique that allows accurate, interpretable and fast classification,” Data Mining and Knowledge Discovery, vol. 22, nos. 1–2, pp. 149–182, 2011.

[169] G. Yuan, P. Sun, J. Zhao, D. Li, and C. Wang, “A review of moving object trajectory clustering algorithms,” Artificial Intelligence Review, vol. 47, no. 1, pp. 123–144, 2017.

[170] J. Zakaria, A. Mueen, and E. J. Keogh, “Clustering time series using unsupervised-shapelets,” in Proceedings of the IEEE International Conference on Data Mining (ICDM), 2012, pp. 785–794.

[171] Y. Zheng, “Trajectory data mining: An overview,” ACM Transactions on Intelligent Systems and Technology, vol. 6, no. 3, pp. 29:1–29:41, 2015.

[172] Y. Cao, W. Hu, L. Li, M. Li, M. Xu, and K. Han, “Bi-STAT+: An enhanced bidirectional spatio-temporal adaptive transformer for urban traffic flow forecasting,” Computers, Materials & Continua, vol. 86, no. 2, pp. 1–23, 2026, doi: 10.32604/cmc.2025.069373.

[173] H. Wang, J. Yong, J. Wang, S. Fukushima, and R. Jiang, “Towards resilient transportation: A conditional transformer for accident-informed traffic forecasting,” arXiv:2512.09398, 2025.

[174] C.-W. Shen, J.-W. Jiang, and H.-P. Hsieh, “STGym: A modular benchmark for spatio-temporal networks with a survey and case study on traffic forecasting,” IEEE Transactions on Big Data, pp. 1–20, 2025, doi: 10.1109/TBDATA.2025.3618482.

[175] Q. Zheng, M. Shao, and Y. Zhang, “TLAST: A time-lag aware spatial-temporal transformer for traffic flow forecasting,” IEEE Transactions on Intelligent Transportation Systems, vol. 26, no. 9, pp. 13144–13158, 2025, doi: 10.1109/TITS.2025.3583391.

[176] R. Ahmed, M. Y. Patel, P. S. Kabyo, M. H. H. Amjad, M. S. S. Shovon, and A. S. M. M. Hasan, “An advanced attention-based neural model for traffic flow forecasting with integrated temporal correlation analysis,” European Journal of Electrical Engineering and Computer Science, vol. 9, no. 3, pp. 1–8, 2025, doi: 10.24018/ejece.2025.9.3.708.

[177] W. Sun, R. Cheng, Y. Jiao, and J. Gao, “Decoupled graph spatial-temporal transformer networks for traffic flow forecasting,” Engineering Applications of Artificial Intelligence, vol. 148, p. 110476, 2025, doi: 10.1016/j.engappai.2025.110476.

[178] L. Wang, J. Zhuang, S. Ma, and H. Lin, “Frequency enhanced dynamic graph convolutional networks for traffic flow forecasting,” IEEE Access, vol. 13, pp. 103451–103462, 2025, doi: 10.1109/ACCESS.2025.3579334.

[179] B.-A. Dai, N. Lyu, and Y. Miao, “FasterSTS: A faster spatio-temporal synchronous graph convolutional network for traffic flow forecasting,” arXiv:2501.00756, 2025..

[180] Y. LeCun, C. Cortes, and C. J. C. Burges, “The MNIST database of handwritten digits,” 1998. [Online]. Available: Yann LeCun’s website.

[181] A. Krizhevsky, “Learning multiple layers of features from tiny images,” Technical Report, University of Toronto, 2009.

[182] Kaggle, "Dogs vs. Cats," Kaggle Competition Dataset, 2013. [Online]. Available: Kaggle competition page.