UNIVERSITY OF TWENTE.

Master Thesis

Forecasting of Generic Long-Lead Items for Engineer-to-Order Production Using TPE-Tuned Deep Neural Networks: A Comparative Evaluation

by

Matthijs de Groot

Industrial Engineering and Management
Specialization Production and Logistics Management
Orientation Supply Chain and Transportation Management
Faculty of Behavioural, Management and Social Sciences

Examination committee
Dr. B.A. Beirigo (Breno)
Dr. M.C. van der Heijden (Matthieu)
University of Twente

External supervision
J. Siau (Johan)
Huisman Equipment B.V.

2025

Preface

Dear reader,

Before you lies the master's thesis: *"Forecasting of Generic Long-Lead Items for Engineer-to-Order Production Using TPE-Tuned Deep Neural Networks: A Comparative Evaluation"*. It has been written as part of the fulfillment requirements for the Master's degree in Industrial Engineering and Management at the University of Twente. The thesis assignment was conducted at Huisman Equipment B.V. in Schiedam.

At Huisman Equipment B.V., I was welcomed by their recruiter, Ewelina Miller, who provided me with the opportunity to pursue a challenging thesis assignment aligned with my interests. She was the one navigating me through the company, and was always willing to assist with any questions or concerns I had. I am grateful for her guidance and support throughout this research.

I am also deeply grateful to my supervisor at Huisman Equipment B.V., Johan Siau, for his invaluable guidance, expertise, and constructive feedback throughout the project. His insights greatly contributed to the quality and depth of this research, and his support was essential in addressing the practical challenges encountered throughout the research.

Moreover, special thanks go to my supervisors from the University of Twente. In particular, I would like to thank my first supervisor, Breno Alves Beirigo, for his consistent support and excellent guidance throughout the project. The regular meetings and in-depth discussions were instrumental in shaping the direction and quality of this thesis. I would also like to thank my second supervisor, Matthieu van der Heijden, for his valuable feedback, which helped steer the research in the right direction.

I would also like to express my gratitude to the Supply Chain team members at Huisman Equipment B.V. for their warm welcome and support throughout my time at the company. Last, I am deeply grateful to my family and friends for their unconditional support and encouragement.

I hope you enjoy readings this thesis.

Matthijs de Groot

Enschede, June 16, 2025

Executive Summary

This master's thesis presents research conducted in partial fulfillment of the requirements for the Master's degree in Industrial Engineering and Management at the University of Twente. The research was carried out at Huisman Equipment B.V., a global heavy lifting and construction equipment manufacturer with expertise in designing, manufacturing, and providing service for customers in the oil and gas, renewables, leisure, and civil industries. Particularly, the study focuses on the New Build business line of Huisman Netherlands (HNL), which is responsible for the realisation of new construction equipment projects.

In line with an Engineer-to-Order (ETO) production strategy, Huisman New Build adopts a customer order decoupling point (CODP) positioned before engineering. The average project lead time equals two years due to the early CODP — resulting in low standardisation — and the high complexity of the projects. This study focuses on the Long-Lead Item (LLI)-driven critical path, as the Supply Chain department suspects that the absence of forecasting leads to missed opportunities for earlier procurement of components with excessive lead times (LLIs), potentially causing significant delays in subsequent phases. To address this issue, this research aims to evaluate demand forecasting models for generic LLIs — defined as those LLIs that are not specifically tailored or custom-made for a particular project, but instead are used across multiple projects over the years — to assess the feasibility of accurately predicting demand in HNL's ETO production environment, potentially enabling earlier procurement. The main research question guiding this study is as follows:

To what extent is it feasible to forecast generic Long-Lead Item (LLI) demand in Huisman Netherlands (HNL)'s Engineer-to-Order (ETO) production environment, and to what extent can this enable earlier procurement decisions?

To address this research question, a structured approach was followed to identify relevant LLIs for earlier procurement and forecasting. First, candidate purchase groups were identified by selecting purchase groups categorised as 'A' in an ABC analysis and excluding those that do not comply with a set of constraints. These constraints ensure adequate data quality and practical relevance within HNL's New Build projects. The Analytical Hierarchy Process (AHP)-express framework was subsequently used to rank the remaining candidate groups, leading to the bearings purchase group. Within this group, a distinction was made between project-specific and generic items using another set of constraints based on their usage across projects, historical demand patterns, and expert interviews. Moreover, generic items which are not considered LLIs were excluded, as their procurement and delivery are typically not in the critical path of the project. Consequently, three generic LLIs were selected with the identifiers LLI-1, LLI-2, and LLI-3. Demand for LLI-1 exhibits neither a statistically significant trend nor seasonality, whereas demand for LLI-2 and LLI-3 show significant linear trends, but no significant seasonality. LLI-1 exhibits a lumpy demand pattern, while historical demands for LLI-2 and LLI-3 are formally classified as erratic but their Coefficient of Variation (CV)^2 and Average Demand Interval (ADI) values are similar to those of LLI-1.

A literature study was conducted to first review existing forecasting techniques, followed by an evaluation of models that have shown good performance in forecasting lumpy demand.

Particularly deep learning models have shown promise in irregular demand forecasting tasks with long time series. However, mixed evidence is available about the relative prominence of Machine Learning (ML) models and statistical models. Therefore, this study includes a broad range of both statistical and ML models: Simple Exponential Smoothing (SES), Double Exponential Smoothing (DES), the Syntetos-Boylan Approximation (SBA), Autoregressive Integrated Moving Average (ARIMA), Long Short-Term Memory (LSTM), Temporal Convolutional Network (TCN), and Transformer. The deep learning models are implemented in both univariate and multivariate variants to assess the value of incorporating external covariate data. A naive Random Walk (RW) model is included as benchmark model.

A nested cross-validation procedure with rolling-origin-recalibration in the outer loop is adopted to validate forecasts over a six-month horizon. As an integral part of this approach, hyperparameters were tuned using the Tree-Structured Parzen Estimator (TPE) on a fixed-size validation set within the inner loop. The Mean Error (ME), Mean Absolute Error (MAE), and Mean Absolute Scaled Error (MASE) were used as evaluation metrics. The models were tested on all available data using time buckets of six months.

Surprisingly, all models struggled to consistently outperform the naive RW benchmark, particularly for LLI-1. For this time series, only SBA outperformed RW in terms of mean out-of-sample MAE. The statistical SBA model clearly outperformed all others — including complex deep learning models like LSTM and Transformer — by achieving the lowest out-of-sample MAE, highest robustness, and minimal bias. SBA achieved a mean ME of -0.830, a mean MAE of 7.178, and a mean MASE of 0.522 (-18.9% compared to the benchmark) out-of-sample. In the time series of LLI-2, the multivariate Long Short-Term Memory (LSTM) model achieved the best generalisation to unseen data, followed by its univariate variant and SBA — these were the only models outperforming the RW benchmark. The multivariate LSTM model achieved a mean ME of 0.108, a mean MAE of 10.981, and a mean MASE of 0.717 (-3.5% compared to the benchmark) out-of-sample on this time series. Last, almost all models outperformed the RW benchmark for LLI-3, with Simple Exponential Smoothing (SES) exhibiting superior results in terms of mean out-of-sample MAE. On average, it achieved a ME of 7.723, a MAE of 19.545, and a MASE of 1.194 (-37.4% compared to the benchmark) out-of-sample. However, we observed substantial variability in forecast accuracy and minor accuracy improvements of the multivariate LSTM compared to the RW benchmark for LLI-2, and high MASE values and substantial variability of SES for LLI-3. LLI-1 is therefore the most forecastable item.

It is recommended for HNL to evaluate project timelines to determine whether early procurement based on forecasts offers practical benefits. If specific items are identified as potentially benefiting from forecast-driven procurement, we recommend integrating the experimental forecasting setup developed in this study into an internal test environment to further explore the feasibility of forecast-based procurement, and adopt a continued focus on improving demand predictability. While ML models did not consistently outperform statistical models in this study, their performance is known to improve with larger and richer datasets, so it is also recommended to prioritise consistent and detailed data collection. If forecast-driven procurement is implemented for selected LLIs, inventory policies should be adapted accordingly. In particular, to effectively manage inventory in forecast-driven procurement, safety stock levels should be calculated based on the MAEs of out-of-sample forecasts.

To improve forecast accuracy and better support future procurement decisions, promising future research directions are recommended to HNL. First, incorporating different combinations of covariate data — such as project-related information or macroeconomic indicators — could enhance deep learning performance. Furthermore, additional model variations — including cross-learning, hybrid, and ensemble models — potentially improve forecast accuracy and robustness.

Contents

List of Figures

List of Tables

List of Acronyms

ADI Average Demand Interval

AHP Analytic Hierarchy Process

AICc Akaike's Information Criterion corrected

ANN Artificial Neural Network

ARIMA Autoregressive Integrated Moving Average

CL Cross-Learning

CNN Convolutional Neural Network

CODP Customer Order Decoupling Point

CCDF Complementary Cumulative Distribution Function

CV Coefficient of Variation

DES Double Exponential Smoothing

ELU Exponential Linear Unit

ETO Engineer-to-Order

HNL Huisman Netherlands

LLI Long-Lead Item

LSTM Long Short-Term Memory

MAE Mean Absolute Error

MAPE Mean Absolute Percent Error

MASE Mean Absolute Scaled Error

ME Mean Error

ML Machine Learning

MLP Multilayer Perceptron

MPSM Managerial Problem Solving Method

MSE Mean Squared Error

PDF Probability Density Function

RW Random Walk

ReLU Rectified Linear Unit

RNN Recurrent Neural Network

SBA Syntetos-Boylan Approximation

SES Simple Exponential Smoothing

SKU Stock-Keeping Unit

sMAPE symmetric Mean Absolute Percent Error

TCN Temporal Convolutional Network

TES Triple Exponential Smoothing

TPE Tree-Structured Parzen Estimator

TSF Time Series Forecasting

Chapter 1

Introduction

This chapter introduces the research conducted at Huisman Equipment B.V. as part of the graduation requirements for the Master of Science in Industrial Engineering and Management at the University of Twente. Section 1.1 provides the research background by introducing the company where the research is conducted. Section 1.2 identifies the problem perceived by Huisman Equipment B.V. and formulates the objective of the research. Section 1.3 presents the research design, including the problem-solving approach (including research questions), research scope, and intended deliverables. The outline of the study is presented in Section 1.4, where the problem-solving phases and corresponding chapters are outlined.

1.1 Background and Context

This research is conducted at the Logistics & Warehouse Group of Huisman Equipment B.V., hereafter referred to as "Huisman" for brevity. Huisman is a family-owned business with extensive experience in designing, manufacturing, and providing service for heavy construction equipment for world-leading companies in the oil and gas, renewables, leisure, and civil industries. Their product portfolio consists of offshore and onshore cranes for heavy lifting, pipelay equipment, drilling equipment, wind turbine installation equipment, rock dumping systems, winches, vessel designs, and specials for example for deep water lowering, skidding and salvage (Huisman Equipment B.V., n.d.-a). Figure 1.1 depicts examples of Huisman-built products (e.g., a tub mounted crane (bottom left), two heavy lift mast cranes (top left), a leg-encircling crane (bottom right), and a multi-lay pipelay system (top right)).

FIGURE 1.1: Huisman Equipment B.V. product portfolio

Delivering step changing technical solutions by constantly working on innovative solutions which add value to the market is the primary goal of Huisman. Their extensive operational experience in realising high-quality heavy construction equipment is used to deliver innovative

solutions for new projects in mechanical, structural, naval, hydraulic, electrical, and software engineering (Huisman Equipment B.V., n.d.-b). Equipment provided by Huisman is often a critical main component, so high-quality solutions and services must be delivered consistently. Huisman aims to become the industry standard by taking innovation, quality, and safety to the next level to provide a competitive edge for their global customer portfolio. The company designs and manufactures custom-built solutions in close collaboration with their customers, so production, testing, commissioning, and installation facilities are located worldwide to deliver their custom-built equipment on time on a turn-key basis. Huisman therefore expanded their engineering and production capacity from their head office in Schiedam, referred to as HNL, to the Czech Republic, China, and Brazil. Additional local sales, commissioning, engineering, service, and after sales support is provided by facilities in Houston and Rosenberg (USA), Bergen (Norway), Enschede (The Netherlands), Perth (Australia), and Singapore.

Huisman consists of three business lines operating in their organisation: Huisman New Build, Huisman Services, and Huisman Geo. Huisman New Build is the business line working on the construction of new heavy construction equipment. Huisman Services is responsible for applying Huisman's expertise to satisfy the service needs of their customers. This includes training, repair advice, repairs, spare part delivery, corrective and preventive maintenance, upgrades, modifications, and re-commissioning. Huisman Geo realises geothermal projects to extract the potential of geothermal heat and power. This research focuses on the New Build business line of HNL.

1.2 Problem Identification

Olhager (2010) defines the Customer Order Decoupling Point (CODP) as the point in the value chain for a product, where the product is linked to a specific customer order. Value adding processes upstream of the CODP are forecast-driven, while processes downstream the CODP are driven by customer orders (van Donk & van Doorne, 2016; Olhager, 2010). The CODP is, by definition, the last stock point along the supply chain (Harfeldt-Berg & Olhager, 2024). The New Build business line of HNL adopts a CODP before engineering. This aligns with an Engineer-to-Order (ETO) production strategy as illustrated in Figure 1.2, where dotted lines represent forecast-driven activities and solid lines indicate customer order-driven processes across different production strategies. As a result, Huisman New Build does not currently hold stock for production components, and all procurement and manufacturing activities are triggered by specific customer orders rather than forecasts.

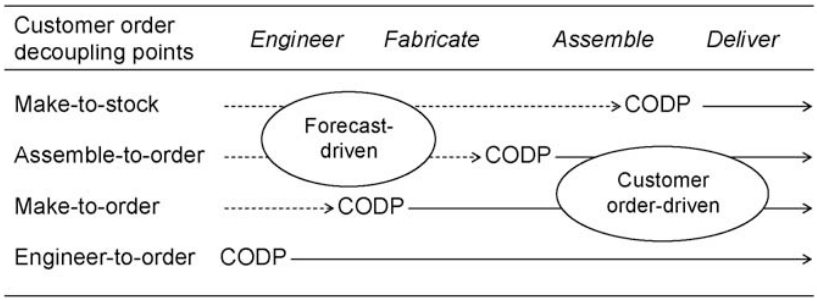


FIGURE 1.2: Illustration of production strategies with their corresponding CODPs (Olhager, 2010)

Additionally, Huisman New Build adopts a concurrent engineering work methodology for its projects, which stimulates the parallelisation of engineering, procurement, and fabrication tasks to shorten the project lead time (see Figure 1.3). However, due to the high complexity of these projects and the customer-driven manufacturing environment, Huisman New Build projects

typically have a lead time of approximately two years. Introducing forecasting for Long-Lead Items (LLIs) offers the potential to shift selected procurement activities upstream of the CODP, making them forecast-driven rather than entirely customer order-driven. This would further support the concurrent engineering approach by enabling earlier procurement decisions, increasing parallelisation, and potentially reducing the overall project lead time.

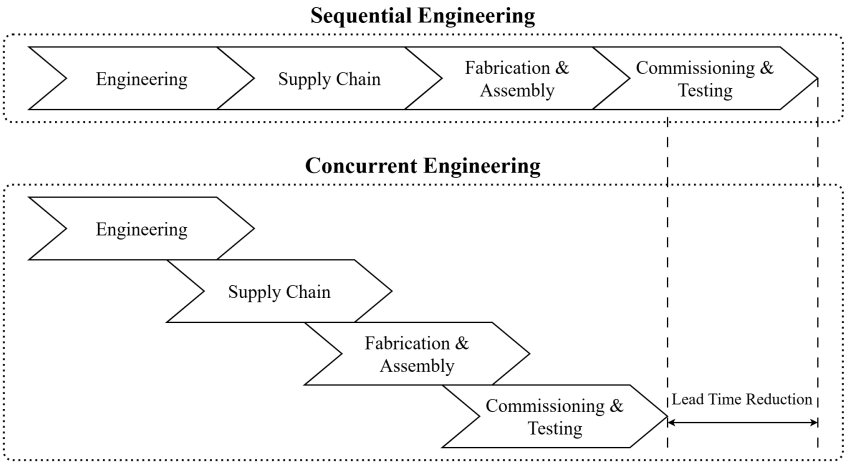


FIGURE 1.3: Illustration of the concurrent engineering methodology adopted by Huisman

This section relates to the first phase of the Managerial Problem Solving Method (MPSM) research framework proposed by Heerkens and Van Winden (2021). The problem identification formulates the problem context by identifying the core problem and research objective.

1.2.1 Problem Statement and Action Problems

To define the problem related to this study, we first map all problems of Huisman with their causal links in a diagram as illustrated in the problem cluster in Figure 1.4. The problem cluster is structured from the bottom up, beginning with the root causes and progressing upward to illustrate the cause–effect relationships. The most downstream problems are defined as action problems, which are discrepancies between the norm and reality, as perceived by the problem owner (Heerkens & Van Winden, 2021). The problem cluster reveals the following action problem:

1. *Missed opportunities.* Huisman's Supply Chain department suspects that the absence of forecasting leads to missed opportunities for earlier procurement of raw materials and components, potentially causing significant delays in subsequent phases, missed contractual deadlines, and increased operational costs.

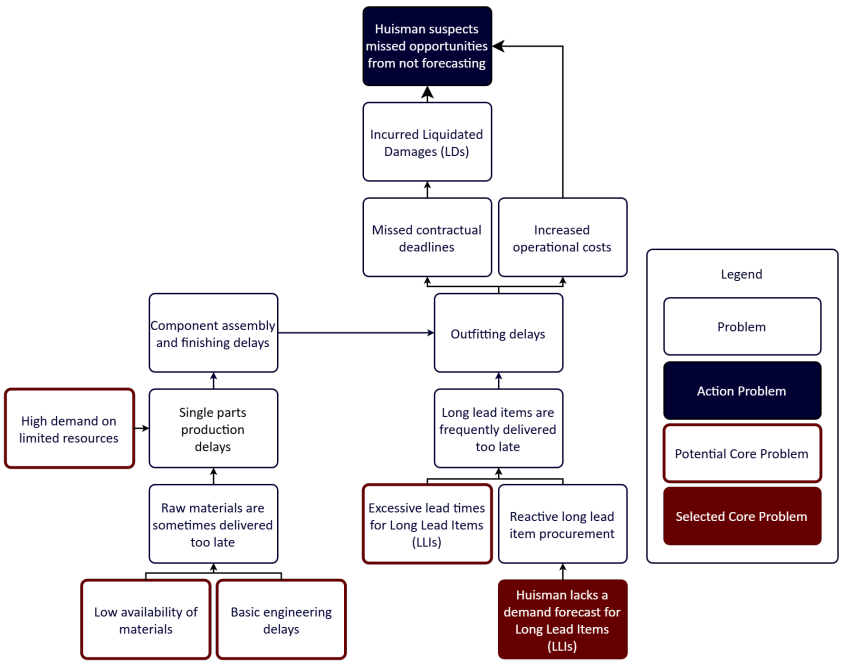


FIGURE 1.4: Problem cluster mapping cause-effect relationships at Huisman

This action problem constitutes the discrepancy between norm and reality as perceived by Huisman. Solving the action problem is the main objective of this research, as Huisman aims to assess the extent of missed opportunities.

1.2.2 Core Problem and Research Motivation

We select the problems which have no direct causes themselves as core problems to prevent solving symptoms of a problem instead of the underlying issue itself (Heerkens & Van Winden, 2021). From the problem cluster in Figure 1.4, we identify the following core problems:

1. *Low availability of materials.* In some cases, low availability of materials — such as steel plates, pipes, shafts, and profiles — at suppliers cause delays in single parts production. This may, in turn, delay assembly, finishing, and outfitting, particularly if the project contains a steel-driven critical path.

2. *Basic engineering delays.* In some cases, delays in the release of basic engineering specifications delay raw material procurement and delivery. The resulting unavailability of raw materials delays single parts production and other downstream manufacturing processes.

3. *High demand on limited resources.* The number of projects at Huisman has significantly increased over recent years, placing increased pressure on available production capacity. This includes both physical resources and skilled personnel required for specialised tasks.

4. *Huisman lacks a demand forecast for Long-Lead Items (LLIs).* Currently, no forecasting model is used to obtain insights into future LLI demand. These items are defined as having replenishment lead times of at least three months, based on expert opinion and historical data from HNL projects. A team of buyers reactively procures LLIs once basic engineering is finished, resulting in frequent late deliveries of LLIs. Approximately 34% of all purchase orders were delivered on or before their planned delivery date over the past four years, 74 days too late on average.

5. *Excessive lead times for LLIs.* LLIs are characterised by complex engineering, highly customised production at the supplier, and long lead times of at least three months, causing their delivery to be a frequent bottleneck of the project execution.

Heerkens and Van Winden (2021) state that it should be possible to influence the selected core problem, and the most important core problem should be selected. In accordance with the problem owner Huisman, we select the following core problem:

Huisman lacks a demand forecast for Long-Lead Items (LLIs).

1.2.3 Research Objective

Given the identified action problem and core problem, we formulate the following research objective to solve the selected core problem:

This research aims to evaluate demand forecasting models for generic LLIs to assess the feasibility of accurately predicting demand in HNL's ETO production environment, enabling earlier procurement.

1.3 Research Design

This section outlines the research design by formulating the main research question and by detailing the problem-solving approach, including the research questions addressed in each phase. Additionally, the scope of the research is listed and intended deliverables to the problem owner are specified.

1.3.1 Problem-Solving Approach

Heerkens and Van Winden (2021) propose their MPSM, a systematic, adaptable problem-solving method applicable for problems encountered in all areas of expertise consisting of seven sequential phases. This research follows this MPSM framework excluding the (final) implementation step: problem definition, approach formulation, problem analysis, solution formulation, solution selection, and evaluation. To guide the execution of each phase after the first two phases, where the global problem and approach are defined, we formulate research questions that systematically address the key knowledge problems within the research framework. The main research question guiding this study is as follows:

To what extent is it feasible to forecast generic LLI demand in HNL's ETO production environment, and to what extent can this enable forecast-driven procurement?

Problem Analysis

The problem analysis phase of the research framework analyses the perceived problem faced by the problem owner by researching the core problem of the research. Chapter 2 presents the problem analysis by providing insights gathered through expert interviews at HNL.

1. What are the current LLI procurement processes at the New Build business line of HNL?
   (a) *How is a typical HNL New Build project currently initiated and structured?*
   (b) *What factors currently determine when LLI procurement can start within a typical New Build project?*

Research question 1 investigates procurement activities within the overall project execution process at HNL New Build. Research question 1a examines the entire project execution process, while research question 1b specifically investigates the LLI procurement process. Analysing these research questions is crucial in understanding the problem and identifying potential points of improvement.

2. Which LLIs are most suitable for earlier procurement through demand forecasting, and what are their historical demand patterns and lead times?

   (a) *What are the most suitable LLIs identified for earlier procurement through demand forecasting in the New Build business line of HNL?*

   (b) *What are the historical demand patterns and average lead times of the identified LLIs?*

Research question 2 is crucial for understanding the factors that are essential for developing a model that aligns with real-world requirements and timelines. It also assesses the feasibility of early procurement and demand forecasting for the identified LLIs at HNL. Research question 2a selects the suitable LLIs for earlier procurement, while question 2b analyses their historical lead times and demand patterns.

3. What are the key requirements for the forecasting model in the HNL use-case?

   (a) *What is the appropriate forecast horizon for the HNL New Build projects, considering procurement lead times and project timelines?*

   (b) *What are the appropriate time buckets for the HNL New Build projects, considering procurement lead times and project timelines?*

   (c) *What is the appropriate aggregation level for the HNL New Build projects, considering procurement lead times and project timelines?*

Research question 3 aims to identify key requirements to set up a forecasting model for the HNL New Build projects. Specifically, it aims to identify the appropriate forecast horizon (3a), time buckets (3b), and aggregation level (3c) necessary to align the forecasting model with real-world procurement decision-making. This ensures the model setup aligns with realistic HNL conditions and delivers practical insights for real-world procurement decisions.

Solution Formulation

This phase formulates alternative solutions and their desirability by identifying best practices for similar problems. It provides the foundation for selecting the most appropriate solution. Solution formulation is addressed in Chapter 3, which comprehensively presents the theoretical background. Chapter 4 builds on this theoretical background by selecting and specifying the most suitable solution methods and experimental design in the HNL use-case.

4. What forecasting models does the literature propose for forecasting demand in the HNL use-case?

   (a) *What demand and forecasting models exist in literature?*

   (b) *Which forecast models have historically shown good performance in similar conditions?*

Research question 4 aims to explore the existing body of literature to identify forecasting models suitable for the selected LLIs. Following research question 4a, a review of academic peer-reviewed journals will be conducted to understand demand and forecasting models. In research question 4b, we research similar use-cases in the current body of literature to identify state-of-the-art alternative solutions.

5. What metrics are proposed in literature to ensure robust forecasting model evaluation?

Research question 5 formulates state-of-the-art evaluation metrics for robust model assessment and selection. This analysis will provide a theoretical foundation to guide the selection of forecasting models tailored to the use-case of HNL.

6. What forecasting models and configurations are selected for comparative evaluation for the HNL use-case?

   (a) *What forecasting models are most suitable for the HNL use-case?*

   (b) *What hyperparameters are identified for the selected models, and how can they be effectively tuned to optimise forecasting model performance for the HNL use-case?*

After having identified state-of-the art forecasting practices in research questions 4 and 5, we formulate the experimental design tailored to the HNL use-case in research question 6. This formulation includes the models used for comparison (6a) and hyperparameter tuning (6b).

7. How can forecasts be evaluated and validated to ensure reliable and accurate model selection for the use-case of HNL?

   (a) *Which forecast accuracy metrics are most suitable for evaluating forecasting models in the HNL use-case?*

   (b) *Which validation schemes are most suitable for validating forecasting models in the HNL use-case?*

Research question 7 further details the experimental design by defining the forecast accuracy metrics (7a) and validation schemes (7b) that will be employed.

Solution Selection

In the fifth phase of the MPSM framework, we select the most suitable solution to the core problem for the use-case of HNL. Key findings and well-founded conclusions of the experimental execution are addressed in the numerical analysis of Chapter 5.

8. Which forecasting model performs best for the selected LLIs in the HNL use-case?

   (a) *How do the selected forecasting models compare in terms of accuracy and robustness for generic LLIs in the HNL use-case?*

   (b) *What are the optimal hyperparameter values identified through hyperparameter tuning for the selected forecasting models?*

   (c) *To what extent does covariate data improve the forecast accuracy of the selected forecasting models in the HNL use-case?*

Research question 8 aims to identify the best-performing and most robust forecasting model in the HNL use-case. The relative performance of the models is presented following research question 8a, while optimal hyperparameters are addressed in research question 8b. Research question 8c focuses on the impact of covariates on the forecasting accuracy.

9. How do the selected models compare in terms of computational complexity?

Research question 10 aims to provide insights into the computational tractability of the selected models. Insights into the computational tractability are crucial to assess the feasibility of applying the models in practice, ensuring they align with the decision-making timelines within the HNL context. It also helps balance forecasting accuracy with practicality - marginal accuracy gains may not justify the use of computationally expensive models.

Evaluation

In the final phase of the MPSM framework, a structured evaluation of all research phases is conducted (Heerkens & Van Winden, 2021). It assesses the extent to which the proposed solution meets the research objective and addresses the core problem. Chapter 6 pertains to this phase, summarising the findings and reflecting on the research process.

10. What are the key findings, implications, and recommendations for the problem owner and further research?

Research question 10 concludes the research by providing holistic, actionable insights for the problem owner. Additionally, it addresses academic contributions and directions for further research.

1.3.2 Research Scope

In this research, we select the purchase group and SKUs that are most suitable for demand forecasting and have the potential to benefit from forecast-driven decision-making in procurement processes. The solution method is specifically designed for the New Build business line of HNL and the conclusions on forecastibility solely apply to the LLIs selected in Section 2.3. Yet, the solution method is generisable to other items. Given that demand patterns may differ, it is recommended to repeat the research to evaluate the most optimal models for each generic item. It is also important to note that this research focuses on assessing the forecastibility of the selected items, procurement or inventory control decision-making optimisation is not included in the scope.

1.3.3 Intended Deliverables

Intended deliverables to the problem owner include:

- A recommendation on the most appropriate demand forecasting model for the selected LLIs, derived from a comparative analysis of alternative models;
- A demand forecasting tool configured to implement the selected forecasting model;
- A comprehensive evaluation of the forecasting performance, based on the application of historical demand data and relevant evaluation metrics;
- Detailed recommendations for the integration of the demand forecasting tool into HNL's operational processes;
- This master's thesis, which will document the research methodology, findings, and evidence-based conclusions aimed at addressing the identified challenges.

1.4 Research Outline

This section outlines the research structure, presenting the sequential MPSM phases of the study, the corresponding chapters in which each phase is addressed, and the research questions that form its foundation. By mapping the research questions to their corresponding phases and chapters, the framework ensures a logical progression throughout the thesis. This clarifies how each part of the study contributes to addressing the overall research objective. Table 1.1 provides an overview of the MPSM phases with their corresponding chapters and research questions. The problem definition and approach formulation phases follow the MPSM methodology and are not driven by specific research questions. For more elaborate descriptions on these phases, refer to Heerkens and Van Winden (2021).

TABLE 1.1: Research Outline

Phase	Chapter	Research questions
1. Problem Definition	1.2. Problem Identification	
2. Approach Formulation	1.3. Research Design	
3. Problem Analysis	2. Context Analysis	1. What are the current LLI procurement processes at the New Build business line of HNL? 2. Which LLIs are most suitable for earlier procurement through demand forecasting, and what are their historical demand patterns and lead times? 3. What are the key requirements for the forecasting model in the HNL use-case?
4. Solution Formulation	3. Literature Study	4. What forecasting models does the literature propose for forecasting demand in the HNL use-case? 5. What metrics are proposed in literature to ensure robust forecasting model evaluation?
	4. Experimental Setup	6. What forecasting models and configurations are selected for comparative evaluation for the HNL use-case? 7. How can forecasts be evaluated and validated to ensure reliable and accurate model selection for the HNL use-case?
5. Solution Selection	5. Numerical Results and Discussion	8. Which forecasting model performs best for the selected LLIs in the HNL use-case? 9. How do the selected models compare in terms of computational complexity?
6. Evaluation	6. Conclusions and Recommendations	10. What are the key findings, implications, and recommendations for the problem owner and further research?

Chapter 2

Context Analysis

This chapter relates to phase 3 of the MPSM framework: Problem Analysis, and — as outlined in Section 1.4 — research questions 1, 2, and 3 are answered. Section 2.1 details all processes in the project execution at HNL New Build. The generic LLIs that are in the scope of this study are selected and specified in Sections 2.2 and 2.3. Moreover, the respective demand patterns are analysed in Section 2.4.

2.1 Project Execution and Procurement Processes

This section describes the current project execution and procurement procedures for Huisman New Build projects and identifies typical bottlenecks in the project planning. The labels in parentheses correspond to activities depicted in Figure 2.1, which visually guides this section by providing a detailed project exectution overview using BPMN 2.0 notation with labelled activities.

2.1.1 Project Initiation and Management

The Tender & Concepts department initiates projects upon receiving a request for quotation from a client. Subsequently, the Concept Engineering team develops a concept design and prepares the associated documentation (TC2). This includes, but is not limited to, user requirements, technical specifications, and an initial weight estimate detailed at the part level. It should be noted that at this stage, the weight estimate is indicative only; it is based on preliminary design assumptions, with no finalised part codes or detailed component definitions established yet. The concept design documentation serves as the basis for aligning with the client on the project scope, specifications, and contract negotiations (TC3-TC5). Upon achieving formal contractual agreement (TC6), the Tender & Concepts department transfers ownership of the project—together with all associated documentation—to the Project Management department (TC7). This department is responsible for the planning, control, and parallel execution of the multidisciplinary project in accordance with the agreed contractual terms and project objectives (PM1-PM4).

2.1.2 Engineering and Supply Chain Processes

After project team kick-off (PM3), post-contract engineering (EN1) verifies the complete availability of concept documentation, thereby supporting engineering activities in accordance with the contractual agreements. Subsequently, system engineering (EN2) defines the formal system requirements, including the identification of critical components (i.e., components in the primary load path, long-lead items, and high-value components). At this stage, exact part numbers or specifications are not yet determined. However, the identified critical components are prioritised for further engineering and specification during basic engineering (EN3), particularly LLIs to mitigate potential risks to the project schedule. After basic engineering, Huisman New Build

adopts a concurrent engineering paradigm by parallelising engineering, supply chain, manufacturing, and field engineering tasks. While the engineering department advances the detailed and manufacturing design specifications (EN4-EN5) and the software production (EN6), the supply chain department uses basic design documentation to establish a procurement control chart (SC1) and initiate sourcing activities (SC2-SC3). At this stage, LLI-procurement is prioritised, followed by the acquisition of raw materials and other components. Last, logistics are coordinated in parallel for all items (SC4-SC5).

2.1.3 Manufacturing Processes and Project Completion

Once the raw material steel for the project has arrived (SC4) and the manufacturing design is completed by engineering (EN5), work preparation can start organising fabrication jobs. These jobs include production, welding, and machining of single parts (M2) and fitting and painting of components (M3). After the completion of these manufacturing processes, outfitting (M4) can begin provided all purchased parts (including LLIs) have been delivered. In this phase, components, single parts, and procured parts are assembled into an assembly ready for installation (FE1) and commissioning and testing (FE2). Control systems are also commissioned and tested once software production (EN6) has completed. When commissioning and testing is completed, project management closes the project (PM5) by handing over the scope to the Huisman Services business line.

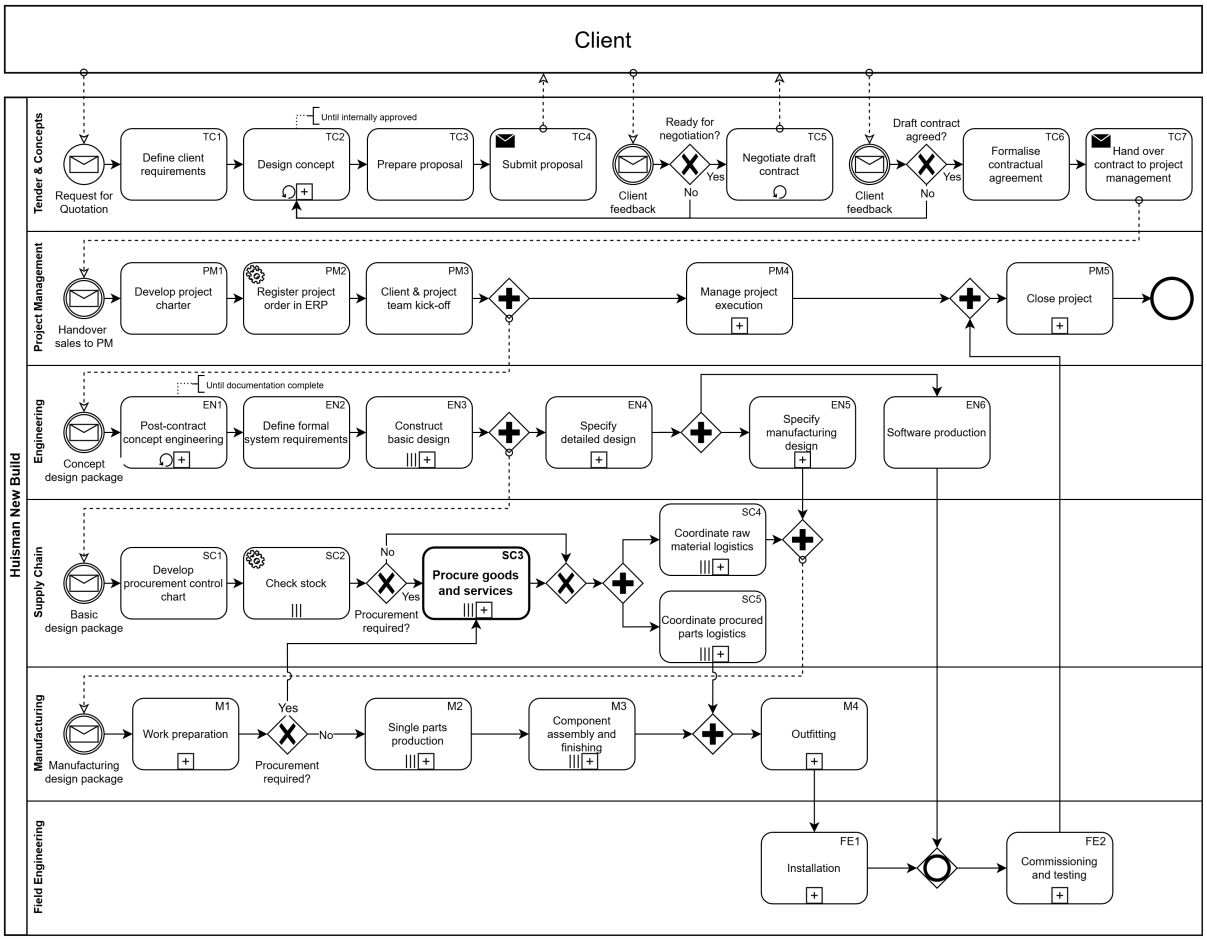


FIGURE 2.1: Detailed overview of Huisman New Build project execution processes across core disciplines.[1] The activity shown in bold (labelled SC3) pertains to the procurement of LLIs and other materials.

2.1.4 Critical Path Activities

We define critical path activities as tasks forming the longest sequence of interdependent activities within a project schedule. These tasks are considered critical because they directly determine the overall project completion date. The Planning Department at Huisman identifies the following general categories of project critical paths:

1. *Steel-driven critical paths* These critical paths arise when the timely availability of raw materials—most commonly steel—is delayed. Such delays may be caused by engineering delays (EN3) or transport disruptions (SC4), subsequently delaying manufacturing activities (M1-M4).

2. *LLI-driven critical paths* LLI-driven critical paths refer to delays or disruptions in a project schedule that arise from the sourcing of LLIs—components (SC3-SC4) that require substantial delivery time due to highly customised production processes at suppliers. Manufacturing, particularly outfitting (M4), is delayed when LLIs are unavailable, which subsequently delays project completion.

3. *Functional critical paths* Fabrication and mobile machining activities (M2-M3) represent the primary critical tasks in functional critical paths. These processes require specialised machinery, which may be unavailable when multiple projects are executed in parallel, potentially resulting in delays to the overall project completion.

2.2 Purchase Group Selection

To ensure data quality and availability, maintain practical relevance, and focus on items with the highest potential for accurate forecasting, we first select a single purchase group as the scope of this research. This focused approach also facilitates better domain understanding, which enables more informed feature engineering and interpretation of forecasting results. This section outlines the methodology used to identify and select the most appropriate purchase group, which forms the foundation for determining the most suitable LLIs for early procurement. We first describe the purchase groups considered as candidates, using an ABC analysis — based on the assumption that high-value purchase groups are more likely to include LLIs due to their complex and specialised nature, and because they often contain items used across multiple projects — and a set of formulated constraints. We then proceed to a comparative evaluation of the candidate purchase groups to identify and select the most suitable purchase group for forecast-driven procurement as the scope of this research.

2.2.1 Purchase Group Identification

Huisman's part archive comprises 287,723 SKUs distributed across 131 distinct purchase groups. These groups primarily include control systems, cut and machined components, electrical, hydraulic, mechanical, and raw material SKUs. The archive also comprises tools, equipment, consumables, and other parts. We use a dataset containing all part issues recorded from July 2003 to November 2024, including, e.g., issue date, part code, quantity, and transaction type. The dataset is filtered to exclude issues from Huisman locations other than HNL. An ABC analysis on the purchase groups, conducted over the past five years, identifies 26 purchase groups categorised as group A, which contribute significantly to the total procurement costs of HNL. These 26 purchase groups constitute approximately 20% of all purchase groups and 84% of the total procurement value of all part issues, as shown in Figure 2.2.

[1]The process is modelled using the standard BPMN 2.0 notation. For further information, see the official specification: https://www.omg.org/spec/BPMN/2.0.2/About-BPMN.

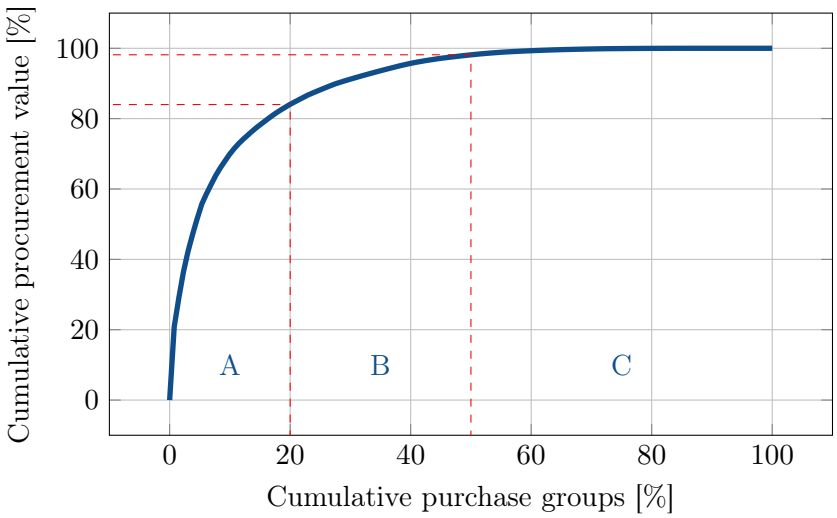


FIGURE 2.2: Distribution-by-value graph of the purchase groups of HNL based on their cumulative procurement value from 2020 to 2024

As it is suggested to eliminate enviable alternatives before ranking (de FSM Russo & Camanho, 2015), we further refine the selection of purchase groups by identifying those that are most relevant to HNL's operations and production processes while ensuring adequate data quality for analysis. From the candidate purchase groups categorised as "A" in the ABC analysis, we exclude those that do not comply with the following set of constraints:

- Purchase groups with SKUs that are easily sourced without impacting project timelines should be excluded, as they do not affect steel-driven, LLI-driven, or functional critical paths, which are crucial to project completion;
- The SKUs within the purchase group must be stored in the HNL warehouse;
- The purchase group must consist exclusively of New Build project-related SKUs;
- A consistent unit of measurement must be used within the purchase group;
- The purchase group must consist of distinct part codes, and grouped part codes (i.e., part codes representing multiple SKUs) should account for no more than 5% of the total transactions in the past five years.

We have identified five candidate purchase groups in group A which comply with all constraints: bearings, gearboxes, hooks, winches, and sheaves. These purchase groups all contain SKUs related to the LLI-driven critical path.

2.2.2 Purchase Group Ranking

To select the purchase group most suitable for forecast-driven procurement, we compare the identified purchase groups using the following set of criteria: data, procurement value, and production criticality. Sub-criteria for the data criterion include data quality and data volume. Unit SKU procurement value and total purchase group procurement value of the part issues over the past five years are sub-criteria of the procurement value criterion. Production criticality is defined as the importance of an SKU in ensuring the timely completion of manufacturing or assembly processes, where its unavailability may cause significant production delays. The procurement lead time and the extent to which the delivery of SKUs within a purchase group are generally located in the critical path of the project timeline are relevant sub-criteria identified for the production criticality.

Multiple Criteria Decision-Making

The AHP is one of the most widely used frameworks to support complex multiple criteria decision-making (Taherdoost & Madanchian, 2023). It is a "theory of measurement through pairwise comparisons and relies on the judgements of experts to derive priority scales" (Saaty, 2008). AHP requires $\binom{m}{2}$ pairwise comparisons per level, given a total of m elements per level. Consequently, applying AHP becomes an arduous task as m increases (Tavana et al., 2023). Tavana et al. (2023) concluded that methods requiring more interaction with experts are less efficient and produce less acceptable results, as experts are more motivated and attentive in methods requiring fewer pairwise comparisons and less interaction. Leal (2020) proposed the simplified AHP-express (best method-AHP), requiring fewer judgement and effort with $m-1$ pairwise comparisons per level. Tavana et al. (2023) found a similar ranking using AHP-express as the traditional AHP in their example, but with far fewer effort and judgement. The AHP-express framework decomposes the decision-making process into the following steps (Tavana et al., 2023):

1. Create a hierarchical tree.
2. Determine the best elements of each level. The elements of each level are compared to other related elements located at a higher level, and the best ones are identified.
3. Determine the preferences of the best element at each level: after interacting with the decision-maker, determine the preferences of the best criterion (best alternative per criterion) relative to the other criteria (using the 9-point scale values of Table A.1), $(a_{Bj}, j = 1, 2, ..., m)$.
4. Calculate the local priorities of the criteria and alternatives per criterion by applying Equation (2.1).
$$w_j = \frac{1/a_{Bj}}{\sum_{k=1}^{m} 1/a_{Bk}}, \quad j = 1, 2, ..., m \tag{2.1}$$
5. Calculate the overall priority of the alternatives.
6. Rank the alternatives.

The AHP-express can be represented in matrix form (Leal, 2020). Table 2.1 provides an overview of the notation of priorities at different levels within the decision hierarchy, which will be used to formulate the matrix representation of AHP-express.

TABLE 2.1: Notation for AHP-express

Symbol	Description
cg_j^i	Priority of sub-criterion j within criterion i
$pasc_{a,j}^i$	Priority of alternative a for sub-criterion j of criterion i
$pac_{a,i}$	Priority of alternative a for criterion i
pc_i	Priority of criterion i for the objective
p_a	Overall priority of alternative a for the objective

For a decision hierarchy with alternatives $a = 1, \ldots, na$ and sub-criteria $j = 1, \ldots, ns_i$ within criterion $i = 1, \ldots, nc$, we determine the local priority vectors $\mathbf{PSC}_i$ for each criterion i and construct an $nc \times \sum_i ns_i$ matrix of local priorities of sub-criteria **MPSC** using Equation (2.2). We then calculate matrix $\mathbf{PASC}_i$ of local priorities of alternative a in each sub-criterion j for each criterion i and group each matrix $\mathbf{PASC}_i$ by each criterion i in an $\sum_i ns_i \times na$ matrix **MPASC**, as shown in Equation (2.3) (Leal, 2020).

$$\mathbf{PSC}_i = \begin{bmatrix} cg_1^i & \cdots & cg_{ns_i}^i \end{bmatrix}, \quad \mathbf{MPSC} = \begin{bmatrix} \mathbf{PSC}_1 & 0 & \cdots & 0 \\ 0 & \mathbf{PSC}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{PSC}_{nc} \end{bmatrix} \tag{2.2}$$

$$\mathbf{PASC}_i = \begin{bmatrix} pasc_{1,1}^i & \cdots & pasc_{na,1}^i \\ \vdots & \ddots & \vdots \\ pasc_{1,ns_i}^i & \cdots & pasc_{na,ns_i}^i \end{bmatrix}, \quad \mathbf{MPASC} = \begin{bmatrix} \mathbf{PASC}_1 \\ \mathbf{PASC}_2 \\ \vdots \\ \mathbf{PASC}_{nc} \end{bmatrix} \tag{2.3}$$

After having constructed matrices **MPSC** and **MPASC**, We multiply the two matrices to obtain an $nc \times na$ matrix **PAC** of local priorities of alternatives a for criteria i, as shown in Equation (2.4). The local priorities of each criterion i are then calculated and used to construct the nc-dimensional row vector **PC** in Equation (2.5) (Leal, 2020).

$$\mathbf{PAC} = \mathbf{MPSC} \cdot \mathbf{MPASC} = \begin{bmatrix} pac_{1,1} & \cdots & pac_{na,1} \\ \vdots & \ddots & \vdots \\ pac_{1,nc} & \cdots & pac_{na,nc} \end{bmatrix} \tag{2.4}$$

$$\mathbf{PC} = \begin{bmatrix} pc_1 & \cdots & pc_{nc} \end{bmatrix} \tag{2.5}$$

We then multiply local priority vector **PC** and matrix **PAC** using Equation (2.6) to obtain a vector of overall priorities of the alternatives for the objective (Leal, 2020).

$$\mathbf{PA} = \mathbf{PC} \cdot \mathbf{PAC} = \begin{bmatrix} p_1 & \cdots & p_{na} \end{bmatrix} \tag{2.6}$$

Last, we acquire the ranking by sorting vector **PA** in descending order such that $p_{a_1} \geq p_{a_2} \geq \cdots \geq p_{a_{na}}$, where a_k denotes the alternative in the k^{th} position of the sorted vector.

Results

In this study, we elicited expert insights from interviews at HNL, applying the 9-point scale to assign relative importance values. For the full operationalisation of the AHP-express framework, including all scoring and matrix calculations, readers are referred to Appendix A. Table 2.2 presents the resulting final ranking for the purchase group selection based on 29 pairwise comparisons using AHP-express. By comparison, AHP would have required a total of $\binom{5}{2} \cdot 6 + \binom{2}{2} \cdot 3 + \binom{3}{2} = 66$ pairwise comparisons. The results indicate that the bearings purchase group is the most suitable option based on the identified criteria, followed by gearboxes, winches, hooks, and sheaves, in descending order of priority. Hence, we select the bearings purchase group.

TABLE 2.2: Final AHP-express ranking of the alternative purchase groups

	Bearings	Gearboxes	Winches	Hooks	Sheaves
p_a	0.338	0.234	0.178	0.136	0.113
Rank	1	2	3	4	5

2.3 Stock-Keeping Unit Selection

In this study, we focus on selecting SKUs (i.e., part codes) within the bearings purchase group that are classified as "generic". We define generic SKUs as those items that are not specifically tailored or custom-made for a particular project, but instead are used across multiple projects over the years. These generic SKU exhibit more stable demand patterns, improving their potential for accurate demand forecasting. We take a pragmatic approach in identifying the generic SKUs by formulating the following set of constraints based on their usage across projects, historical demand patterns, and expert interviews:

- Part codes starting with "9" are excluded, as they represent dummy part codes used for multiple distinct SKUs;
- The item must be used in more than 40 distinct projects over the past 20 years, ensuring it is not project-specific;
- To ensure recent use, items must have at least one non-zero demand occurrence in both 2023 and 2024;
- We only include items with an ADI less than 4 and a squared CV of non-zero demand sizes of less than 5 to filter out highly unstable demand patterns.

Out of the 655 bearings identified as being used historically for HNL New Build projects, only four were selected as generic bearings. This reflects the ETO approach at Huisman, where a broad range of bearings with varying technical specifications are required. As a result, the distribution of monthly non-zero demand occurrences per SKU within the bearings purchase group, from 2004 to 2024, is highly right-skewed (we do not include this empirical distribution here for the sake of conciseness, the corresponding cumulative density histogram can be found under Appendix B). Moreover, we exclude generic bearings which are not considered LLIs since these bearings are generally not in the critical path of the project (see Section 2.1.4). The remaining candidate LLIs — 2002173, 2005480, and 2008960 — all have an average lead time of approximately five months. The nomenclature of the resulting three identified generic LLIs, including their part codes and descriptions, is provided in Table 2.3. For brevity, the selected LLIs 2002173, 2005480, and 2008960 are hereafter referred to as LLI-1, LLI-2, and LLI-3, respectively.

TABLE 2.3: Nomenclature of the identified generic LLIs

Notation	Part Code	Description	Dimensions
LLI-1	2002173	Cylindrical Roller Bearing	100 × 150 × 67
LLI-2	2005480	Cylindrical Roller Bearing	220 × 300 × 95
LLI-3	2008960	Cylindrical Roller Bearing	140 × 200 × 80

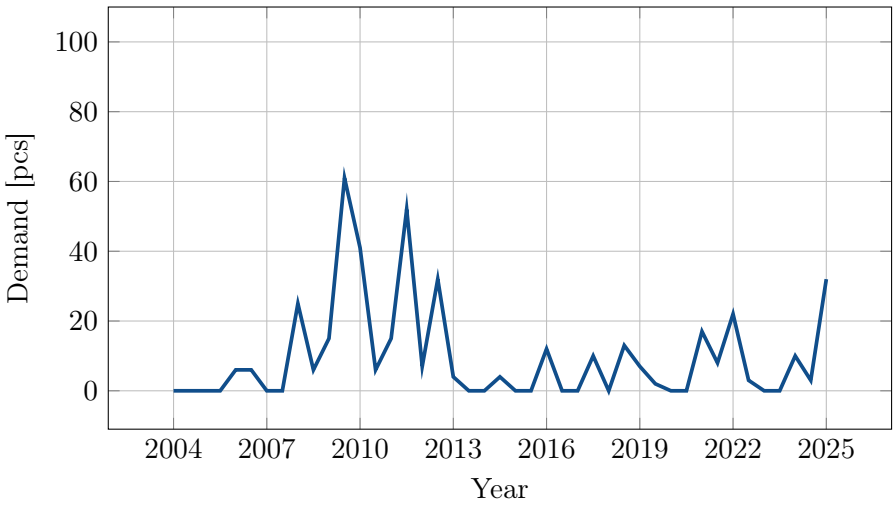
Note: Dimensions are denoted as inner diameter (d) × outer diameter (D) × width (B) [mm].

2.4 Forecasting Requirements and Demand Patterns

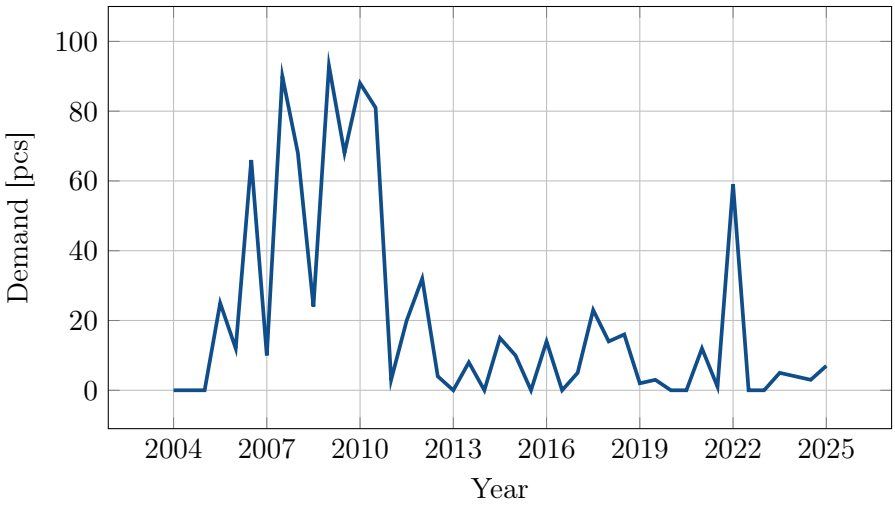
This section defines the forecasting requirements — including time buckets, forecast horizon, and aggregation level — for the presented use case. Moreover, historical demand patterns of the identified generic LLIs are evaluated in this section. Appropriate demand models are selected for each item using statistical tests, and demand patterns are classified to provide context for the forecasting task.

2.4.1 Forecasting Requirements and Historical Demand

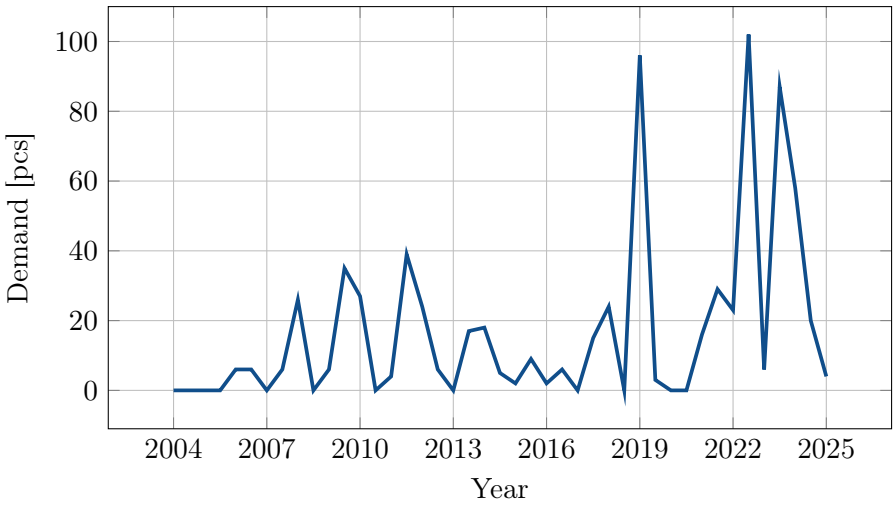
To enable LLI procurement before basic engineering has completed, we require an aggregation level at SKU-level, as procurement decisions are made for specific part codes rather than for aggregated groups. We use a six-month forecast horizon, comprising the five-month lead times of the generic LLIs and an additional one-month review period, as required by the problem owner, to ensure the forecasts are actionable for procurement decision-making. Six-month time buckets are used, as we are interested in the total demand over the forecast horizon and do not require more granular detail. Figure 2.3 depicts the historical demands of LLI-1, LLI-2, and LLI-3, aggregated into six-month time buckets.



(A) Historical Demand for LLI-1



(B) Historical Demand for LLI-2



(c) Historical Demand for LLI-3

FIGURE 2.3: Historical demand time series for the selected LLIs.

2.4.2 Demand Models

Axsäter (2015) describes the constant demand model as the starting point in modelling underlying demand patterns. Demands x_t in period t are represented by random deviations ε_t from a relatively stable average a in the constant model. These random deviations ε_t cannot be forecast and are assumed to be independent with a mean equal to zero. The trend demand model extends the constant demand model by including a systematic linear trend using a variable b representing the systematic increase or decrease per period. If a product's demand exhibits seasonality, the trend-seasonal model is used to capture its demand pattern. If we let F_t denote the seasonal component — a fixed value in the additive model or a scalar multiplier in the multiplicative model — we obtain additive and multiplicative forms of the trend-seasonal model, respectively. For a more comprehensive description of demand models, readers are referred to Axsäter (2015).

To determine a suitable forecasting technique, we need to have some idea of how to model the stochastic demand (Axsäter, 2015). We evaluate the presence of trend and seasonality in the historical demand of the three identified LLIs to select appropriate demand models.

Assessing the Presence of Trend

Trend significance is assessed by first deseasonalising the demand data, followed by performing a linear regression t-test. The fitted regression lines for each LLI are shown in Appendix C. The slopes of the fitted linear regressions are -0.273 for LLI-1, -1.286 for LLI-2, and 0.799 for LLI-3, reflecting negative slopes for the regression lines of LLI-1 and LLI-2, and a positive slope for that of LLI-3. To assess whether these slopes indicate statistically significant linear trends, we formulate the following null hypothesis H_0 and alternative hypothesis H_1 for the linear regression t-test:

H_0: There is no statistically significant linear trend in the deseasonalised data, i.e., $\beta_1 = 0$.

H_1: There is a statistically significant linear trend in the deseasonalised data, i.e., $\beta_1 \neq 0$.

Table 2.4 presents the resulting p-values for the identified LLIs. For LLI-2 and LLI-3, we reject the null hypothesis H_0 at a significance level of 5%, as their p-values are smaller than 0.05. For LLI-1, we fail to reject H_0, as its p-value is greater than 0.05. The linear regression t-test

therefore indicates that LLI-2 and LLI-3 exhibit a significant linear trend, while LLI-1 does not show a statistically significant trend in its deseasonalised demand data.

TABLE 2.4: p-values of the trend significance t-test for the identified generic LLIs

	LLI-1	LLI-2	LLI-3
p-value	0.174	3.67e-04	0.014

Assessing the Presence of Seasonality

A similar approach is taken for assessing the presence of monthly seasonality in the underlying demand patterns: First, we detrend the data to obtain stationary series. Then, we regress the detrended series on time using one dummy variable for the first six-month period of each year representing structural differences relative to the second six-month period. A t-test is used to assess whether the estimated seasonal effect is statistically significant. Since only one dummy variable is included, an overall F-test for joint significance of multiple seasonal factors is not required. The hypotheses for the t-test can therefore be formulated as:

H_0: Seasonal factors do not significantly reduce the variability of the detrended data.

H_1: Seasonal factors significantly reduce the variability of the detrended data.

All individual t-tests result in p-values exceeding the 5% significance level, as depicted in Table 2.5. Consequently, we fail to reject the null hypothesis H_0 for all selected LLIs. We conclude that there is no statistically significant evidence of monthly seasonal patterns in the detrended demand series for any of the identified generic LLIs.

TABLE 2.5: T-test p-values for the seasonal dummy variable for selected generic LLIs.

	LLI-1	LLI-2	LLI-3
p-value	0.968	0.920	0.709

In summary, demand for LLI-1 exhibits neither a statistically significant trend nor seasonality. The historical demand patterns for the other generic LLIs — LLI-2 and LLI-3 — show significant linear trends, with a downward trend for LLI-2 and an upward trend for LLI-3, but no significant seasonality was found. Accordingly, the constant demand model is selected for part code LLI-1, while the trend demand model is chosen for LLI-2 and LLI-3.

2.4.3 Demand Classification

To complement the demand model identification, we further classify the historical demand patterns of the identified generic LLIs using the categorization framework proposed by Syntetos et al. (2005). This approach uses threshold values (ADI= 1.32 and CV^2= 0.49, where ADI is the average interval between non-zero demands and CV^2 is the squared coefficient of variation of non-zero demand sizes) for categorisation into four types: smooth, intermittent, erratic, or lumpy. Demand data are aggregated into 6-month time buckets to calculate the classification metrics. Figure 2.4 shows a plot of all SKUs in the bearings purchase group mapped according to their CV^2 and ADI values. The red dashed lines indicate the classification thresholds as proposed by Syntetos et al. (2005). The majority of SKUs (484 in total) show intermittent behaviour (ADI $>$ 1.32 and $CV^2 \leq 0.49$), while 146 are classified as lumpy (ADI $>$ 1.32 and $CV^2 > 0.49$). Additionally, 12 SKUs exhibit smooth demand patterns (ADI $\leq$ 1.32 and $CV^2 \leq 0.49$) and five are classified as erratic (ADI $\leq$ 1.32 and $CV^2 > 0.49$). Notably, LLI-1 shows a lumpy demand pattern, indicating highly variable non-zero demand sizes and infrequent demand occurrences.

Although historical demands for LLI-2 and LLI-3 are formally classified as erratic, their CV^2 and ADI values are similar to those of LLI-1 as depicted in Figure 2.4.

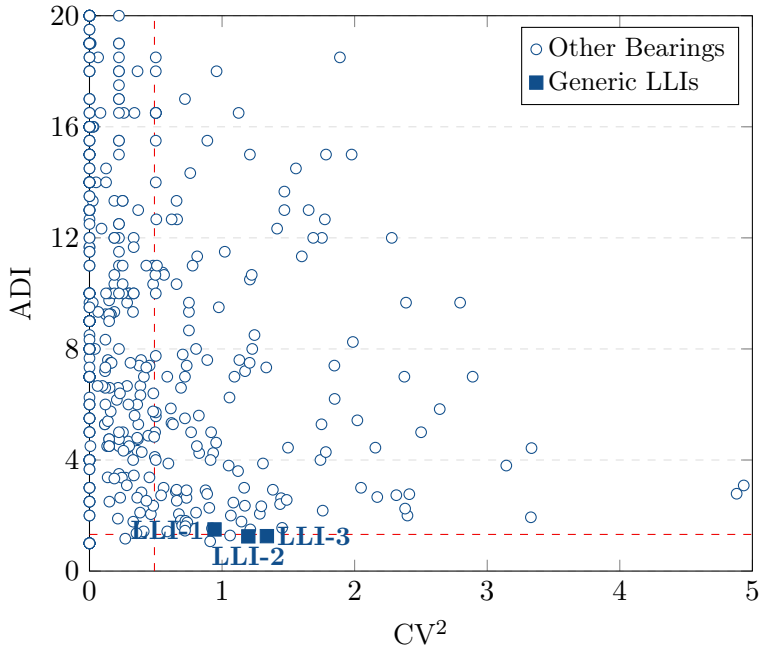


FIGURE 2.4: Plot of CV^2 vs ADI for the demand classification of SKUs in the bearings purchase group. The horizontal and vertical dashed red lines represent the cutoff values used for the ADI and CV^2, respectively.

2.5 Findings and Implications

This chapter contributes to Phase 3: Problem Analysis of the research framework. Current practices at HNL are assessed, detailing the context and requirements of the forecasting task. It addresses research questions 1, 2, and 3, thereby concluding Phase 3 of the MPSM.

1. What are the current LLI procurement processes at the New Build business line of HNL?

Section 2.1 provides a detailed overview of the end-to-end project execution processess, providing key insights into operational characteristics that influence procurement decisions. Key findings include:

- Although engineering and procurement efforts generally prioritise LLIs, procurement cannot begin until after its basic design finished. During concept and system design phases, no exact part codes are released for procurement.
- Typically, project critical paths can be classified as: Steel-driven, LLI-driven, and functional critical paths (see Section 2.1.4). Timely procurement of steel and LLIs is therefore a critical activity in the project execution, and optimising these processes might significantly reduce overall project lead time.

2. Which LLIs are most suitable for earlier procurement through demand forecasting, and what are their historical demand patterns and lead times?

This chapter describes the selection process. Within the bearings purchase group, we identify part codes 2002173 (LLI-1), 2005480 (LLI-2), and 2008960 (LLI-3) as generic LLIs. LLI-1 exhibits a lumpy demand pattern, while historical demands for LLI-2 and LLI-3 are formally classified as erratic but their CV^2 and ADI values are similar to those of LLI-1. LLI-1 follows

a constant demand model, while LLI-2 and LLI-3 are best characterised by a trend demand model. All three SKUs have an average lead time of 5 months.

3. What are the key requirements for the forecasting model in the HNL use-case?

Section 2.4.1 describes the forecasting requirements. Key requirements include:

- We require an aggregation level at SKU-level.
- A forecast horizon of six months — comprising the five-month average lead time and an additional one-month review period — should be used.
- We require six-month time buckets for the historical demand data.

Chapter 3

Literature Study

This chapter provides the theoretical framework of this study by outlining current best practices in demand forecasting. General forecasting techniques are reviewed first, followed by an evaluation of approaches specifically suited to the characteristics of the presented case study. The theoretical framework established in this chapter contributes to phase 4 of the MPSM framework — Solution Formulation — by answering research questions 4 and 5. Section 3.1 presents the concepts of demand forecasting models in existing literature. Moreover, forecast evaluation metrics are detailed in Section 3.2, and Section 3.3 outlines recent studies with characteristics similar to the presented use-case to identify state-of-the-art solutions.

3.1 Forecasting Models

Forecasting models can roughly be classified as judgemental (qualitative) forecasting, time series (extrapolative) forecasting, causal forecasting, and Machine Learning (ML) forecasting (Ali et al., 2009; Archer, 1980; Liang et al., 2024; Barbosa et al., 2015). Additionally, ensemble forecasting combines information and pooles errors by combining outputs from different models and data from different sources (Wu & Levinson, 2021). Hybrid forecasting integrates conventional statistical models with advanced ML models to achieve more comprehensive and reliable forecasting (Liang et al., 2024). This section explains the concepts of judgemental forecasting, Time Series Forecasting (TSF), causal forecasting, and ML forecasting and presents widely applied forecasting models for each forecasting type.

3.1.1 Time Series Forecasting Models

Time series, a class of temporal data objects, is a collection of observations in chronological order (Fu, 2011). Let $Y = \{y_1, ..., y_n\}$ denote a time series. TSF denotes the process of estimating the future values of Y, y_{n+h}, where h denotes the forecasting horizon (Cerqueira et al., 2019). It assumes future trends will be similar to historical trends and therefore utilises statistical methods to capture historical trends to predict future outcomes. Extrapolation of historical data is the most common approach to obtain forecasts over a short horizon. Typical for demand forecasting in connection with inventory control is that it is generally not necessary to implement a time horizon of more than one year (Axsäter, 2015).

Quantitative TSF methods can be grouped into two categories: univariate and multivariate. Univariate methods refer to approaches that model future observations according to a time series that consists of single (scalar) past observations recorded sequentially over equal time increments (National Institute of Standards and Technology, n.d.). Multivariate time series methods are considered extensions of univariate time series approaches as they consider additional time series that are used as explanatory variables (Cerqueira et al., 2019). Univariate and multivariate TSF methods assume that the basic stochastic process has a certain structural

formation which may be described by a small number of parameters, making them relatively easy to implement using traditional statistical models ensuring transparency and limited computational requirements (Gautam & Singh, 2020).

TSF is very suitable in the case of Huisman as it is the most common and important approach to obtain forecasts for the typically short time horizon of demand forecasting in inventory control and it is relatively easy to interpret and apply in computerised inventory control systems (Makridakis et al., 2018). Additionally, TSF can easily update forecasts for thousands of items, which is useful in connection with practical inventory control (Axsäter, 2015) and extensive univariate time series data on LLI demand is available.

Simple Exponential Smoothing

SES, or exponentially weighted moving average, uses exponentially decreasing weights over time. A smoothing factor $\alpha \in [0, 1]$ is used to determine how quickly the weights of past observations decrease exponentially. SES is used to estimate the parameter a for the constant demand model, as the model is not used when there is a pronounced trend or seasonality (Gardner, 2006; Ostertagová & Ostertag, 2011). This means that the forecast for any $\tau > t$ is again the same. SES is a univariate TSF model. Formally, the updating procedure of SES is formulated as (Brown & Meyer, 1961):

$$\hat{x}_{t,\tau} = \hat{a}_t = (1 - \alpha)\hat{x}_{t-1} + \alpha x_t \tag{3.1}$$

Rewriting (3.1) yields:

$$\hat{x}_{t,\tau} = \hat{a}_t = \alpha \sum_{k=0}^{t-1} (1 - \alpha)^k x_{t-k} \tag{3.2}$$

From (3.1), it is apparent that a weight of $1 - \alpha$ is applied to the most recent forecast, while a weight of α is assigned to the most recent observation. A smoothing factor $\alpha = 0$ therefore implies that we ignore the most recent observation and do not update our previous forecast, while $\alpha = 1$ indicates that we take the most recent demand observation as our forecast (Axsäter, 2015). There are no consistent guidelines on the selection of the value of the smoothing factor α (Ravinder, 2013). However, ad hoc values of α in the range 0.1 to 0.3 are typical in the forecasting literature (Gardner Jr, 1985; Montgomery et al., 1990; Barrow et al., 2020). However, nowadays it is common practice to estimate the value of α by minimising the Mean Squared Error (MSE) (Barrow et al., 2020; Petropoulos et al., 2022). When starting to forecast in period t, an initial forecast to be used as $\hat{x}_{t-1}$ is needed as we see in Equation (3.1). Some simple average of the average period demand can be used for this. If such an average cannot be determined, it is possible to start with $\hat{x}_{t-1}$, as this will not affect the forecast in the long run (Axsäter, 2015).

Equation (3.2) clearly shows that the weights $\alpha, (1-\alpha), (1-\alpha)^2, ..., (1-\alpha)^{t-1}$ are used. Therefore, $\hat{x}_{t,\tau}$ is the exponentially weighted moving average of all past observations because the weights decline towards zero exponentially (Ostertagová & Ostertag, 2011). Refer to Table 3.1 for an overview of the notation used in exponential smoothing techniques.

TABLE 3.1: Notation for exponential smoothing

Symbol	Description
x_t	Observed value of the time series in period t
a	Level, i.e., baseline demand per period t
b	Trend, i.e., systematic increase or decrease per period t
F_t	Additive or multiplicative seasonal index in period t
$\hat{x}_{t,\tau}$	Forecast for period $\tau > t$ after observing demand in period t
$\hat{a}_t$	Estimate of a after observing demand in period t
$\hat{b}_t$	Estimate of b after observing demand in period t
$\hat{F}_t$	Estimate of F after observing demand in period t
α	Smoothing factor for the level of the series
β	Smoothing factor for the trend of the series
γ	Smoothing factor for the seasonal factors

Double Exponential Smoothing

Double Exponential Smoothing (DES), or exponential smoothing with trend, was introduced by Holt (1957) and extends SES by assuming that demand follows the trend demand model. Since we are extending SES by including trend, both a and b need to be estimated (Gardner, 2006). These estimates $\hat{a}_t$ and $\hat{b}_t$ are successively updated according to (3.3) and (3.4) using smoothing factors $\alpha \in [0, 1]$ and $\beta \in [0, 1]$ for the level and trend, respectively (Chopra & Meindl, 2007; Axsäter, 2015).

$$\hat{a}_t = (1 - \alpha)(\hat{a}_{t-1} + \hat{b}_{t-1}) + \alpha x_t \tag{3.3}$$

$$\hat{b}_t = (1 - \beta)\hat{b}_{t-1} + \beta(\hat{a}_t - \hat{a}_{t-1}) \tag{3.4}$$

The estimates $\hat{a}_t$ and $\hat{b}_t$ correspond to the period t in which we just observed demand. DES applies a linear trend to forecast demand for a future period $t + k$. For the k-ahead period, we obtain:

$$\hat{x}_{t,t+k} = \hat{a}_t + k\hat{b}_t \tag{3.5}$$

In contrast to the SES model (3.1), DES (3.5) results in different forecasts for future periods. Both updates (3.3) and (3.4) use a weighted average of the observed value and the old estimate (Chopra & Meindl, 2007). Note that Equation (3.3) is essentially equivalent to the constant demand model, as both equations update the best estimate for $\hat{a}_t$ in each period. Equation (3.3) includes trend and considers $\hat{a}_{t-1} + \hat{b}_{t-1}$ as the best estimate for the mean demand in period t. In (3.4), we use $\hat{a}_t - \hat{a}_{t-1}$ as to update the trend, because the average difference between all two consecutive values of $\hat{a}_t$ should be equal to the trend (Axsäter, 2015). The linear regression (3.5) then uses the updated values for the level and trend to provide a forecast for period $t + k$.

DES uses smoothing factors α and β to assign weights to the most recent observation and the old estimate for the level and trend, respectively. Axsäter (2015) recommends a relatively low value for β since errors in the trend may result in significant forecast errors for relatively long forecast horizons since the trend is multiplied by k in (3.5). Typical values for monthly updates are $\alpha = 0.2$ and $\beta = 0.05$. A reasonable initial value for $\hat{a}_t$ is to use some estimate of the mean demand per period. For the trend, we generally use an initial trend of 0 (Axsäter, 2015).

Triple Exponential Smoothing

Triple Exponential Smoothing (TES) is a generalisation of DES since it includes both trend and seasonality in the forecast (i.e., TES can forecast demand for the additive and multiplicative

trend-seasonal models). It is also referred to as exponential smoothing with trend and seasonality or the Holt-Winters method for additive or multiplicative seasonality. In line with SES and DES, TES is used for univariate TSF. Recall from Section 2.4 that no statistically significant seasonality was observed in the historical demand patterns of the generic LLIs. We therefore exclude TES.

Croston's Method

Croston (1972) shows that using exponential smoothing methods with non-normal (sporadic) demand patterns almost always produce inappropriate stock levels in stock control systems. Dealing with intermittent demand is challenging because it involves irregular observations in time and many zero values. However, intermittent time series have attracted considerable attention in forecasting literature (Jeon & Seong, 2022). Croston's TSF method proposed by Croston (1972) has proven to be superior to other exponential smoothing methods for intermittent datasets (Syntetos, 2001; Willemain et al., 1994).

Let as before x_t denote the observed demand in period t and let $\alpha \in [0, 1]$ denote the smoothing factor for the average interval between positive demands and the average size of the positive demand. Schultz (1987) proposed an additional smoothing parameter β to separate the smoothing factors for the average interval between positive demands and the average size of the positive demand. However, this slight modification has not been widely adopted and is therefore disregarded in this thesis (Syntetos, 2001). The estimated number of periods between two positive demands $\hat{k}_t$ and the estimated average size of a positive demand $\hat{d}_t$ are updated in case of a positive demand to forecast the demand $\hat{a}_t$ for period t. The variable k_t is used to update $\hat{k}_t$ and is defined as the stochastic number of periods between two positive demand (Axsäter, 2015). Refer to Table 3.2 for the complete notation used in Croston's method.

Table 3.2: Notation for Croston's method

Symbol	Description
x_t	Observed value of the time series in period t
k_t	Stochastic number of periods since the preceding positive demand of period t
$\hat{k}_t$	Estimated average number of periods between two positive demands at the end of period t
$\hat{d}_t$	Estimated average size of a positive demand at the end of period t
$\hat{a}_t$	Estimated average demand per period at the end of period t
α	Smoothing factor for the average interval between positive demands and the average size of the positive demand

The updating procedure proposed by Croston (1972) is formally stated as follows:

(i) If $x_t = 0$:

$$\begin{aligned} \hat{k}_t &= \hat{k}_{t-1} \\ \hat{d}_t &= \hat{d}_{t-1} \end{aligned} \tag{3.6}$$

(ii) If $x_t \in \mathbb{Z}^+$:

$$\begin{aligned} \hat{k}_t &= (1-\alpha)\hat{k}_{t-1} + \alpha k_t \\ \hat{d}_t &= (1-\alpha)\hat{d}_{t-1} + \alpha x_t \end{aligned} \tag{3.7}$$

We obtain the following forecast for the demand per period:

$$\hat{a}_t = \hat{d}_t \ / \ \hat{k}_t \tag{3.8}$$

Typical values $\alpha \in [0, 0.3]$ tend to be used in practice for Croston's method (Boylan & Syntetos, 2007). Syntetos (2001) found $\alpha = 0.05$ to be optimal for the particular dataset. However, there is no universally optimal value for α.

Despite the theoretical superiority of Croston's method for intermittent demand forecasting, empirical evidence suggests modest improvements in forecast accuracy with respect to basic forecasting models; some evidence even suggests losses in performance (Syntetos & Boylan, 2001). Levén and Segerstedt (2004), among others, proposed a modified Croston method. This modified Croston method was recommended because of improved performance and practicality. However, Boylan and Syntetos (2007) identified a statistical weakness of the modified Croston and state that Croston's method is generally more accurate than its modification. Syntetos and Boylan (2001) identified a mistake in Croston's mathematical derivation of the expected estimate of demand, contributing to the forecast bias. A modification in Croston's method was developed to theoretically eliminate the forecast bias. The bias was empirically tested and successfully corrected by multiplying the demand per period with a factor $(1 - \alpha/2)$ (Syntetos & Boylan, 2001). We refer to this modification as the Syntetos-Boylan Approximation (SBA).

Box-Jenkins Techniques

The demand models from Section 2.4.2 assume independent stochastic variations ε_t. However, situations exist with positively correlated demand or negatively correlated demand (Axsäter, 2015). Box and Jenkings (1970) have developed forecasting models for correlated stochastic demand variations and other general demand processes. These non-seasonal models are known as Autoregressive Integrated Moving Average (ARIMA). Common notation for ARIMA is ARIMA(p, d, q). Here, p is the order of the autoregressive part, d is the degree of first differencing involved, and q represents the order of the moving average part (Axsäter, 2015). Many specific models (e.g., moving average models, exponential smoothing, autoregressive models) are special cases of the general ARIMA model (Gilbert, 2005). For instance, SES and DES are denoted ARIMA(0,1,1) and ARIMA(0,2,2), respectively. The full ARIMA(p, d, q) model, with $x_t^{'}$ defined as the differenced series (which may have been differenced more than once), can be expressed as follows (Hyndman & Athanasopoulos, 2018):

$$x_t^{'} = a + \phi_1 x_{t-1}^{'} + ... + \phi_p x_{t-p}^{'} + \theta_1 \epsilon_{t-1} + ... + \theta_q \epsilon_{t-q} + \epsilon_t \tag{3.9}$$

We define ϕ and θ as the autoregressive parameter and moving average parameter, respectively. For more complicated ARIMA models, the backshift notation is generally preferred. This notation is obtained by defining backshift operator $B^p = x_{t-p}$ and rewriting Equation (3.9):

$$\underbrace{(1 - \phi_1 B - ... - \phi_p B^p)}_{\text{AR}(p)} \ \underbrace{(1 - B)^d x_t}_{d \text{ differences}} \quad = \quad a + \underbrace{(1 + \theta_1 B + ... + \theta_q B^q)\epsilon_t}_{\text{MA}(q)} \tag{3.10}$$

Obtaining forecasts using ARIMA is done by iteratively increasing the period and following three steps for each period until all forecasts are calculated (Hyndman & Athanasopoulos, 2018).

1. Expand Equation (3.10) so that x_t is on the left side.
2. Rewrite the resulting equation by replacing t with $t + k$.
3. Replace future observations with their forecasts, future errors with zero, and past errors with the corresponding residuals on the right side of the equation.

Determining appropriate values for p, d, and q can be complex. Axsäter (2015) states that it is generally sufficient to consider $p, d, q \in \{0, 1, 2\}$ in practice. This will simplify identifying the optimal model while still covering a large set of models. However, functions in Python and R exist to determine appropriate values automatically (Cerqueira et al., 2019; Hyndman & Athanasopoulos, 2018).

3.1.2 Judgemental Forecasting Models

Judgemental (qualitative) forecasting is commonly used in management science literature if historical data is scarce due to situations without historical precedents such as newly launched products (Mishra et al., 2022). Additionally, judgemental forecasting plays an important role in demand forecasting where generally known or anticipated effects of environmental factors on the demand can be incorporated in the forecasts (M. Lawrence et al., 2006) and in macroeconomic forecasting (Fildes & Stekler, 2002). Fildes and Hastings (1994) state that subjective forecasting techniques based on expert opinion are applied more widely than quantitative forecasting techniques in practice despite scepticism of the researcher towards qualitative forecasting in the last decades. However, judgemental forecasting is increasingly accepted in the past few years and its techniques are generally implemented in three different ways. First, it is used when quantitative methods are inapplicable and infeasible as there is no historical data. Second, when data is available, and forecasts from quantitative forecasting methods are tweaked using judgemental insights. Last, when judgemental forecasting is used separately from quantitative models to incorporate the results from both in a final forecast (Mishra et al., 2022). Therefore, it is generally implemented in combination with quantitative methods if time series data is available. Zellner et al. (2021) and M. J. Lawrence et al. (1986) also state that neither TSF nor judgemental forecasting is universally superior, but the two can complement each other for more accurate forecasting. However, Makridakis et al. (1993) found few or no differences in forecast accuracy in the presence of judgemental insights. Additionally, extensive domain knowledge is required in qualitative forecasting, making it expensive and time-consuming. Another limitation regards the subjective nature of judgemental forecasting, influencing the reliability of the forecasts (Mishra et al., 2022). Demand forecasting for inventory control typically concerns a relatively short time horizon at microeconomic level according to Axsäter (2015). Additionally, extensive univariate time series data on LLI demand is available at Huisman. Judgemental forecasting is therefore excluded from this research.

3.1.3 Causal Forecasting Models

Conventional univariate TSF models generally neglect potential causal relationships between the dependent variable and independent variable (Luo et al., 2024). In the presence of causal relationships, historical data can become unrepresentative for future values (Axsäter, 2015). Causal forecasting models incorporate the relationship between the dependent variable and one or multiple independent variables in quantitative models. Variables that are believed to be the drivers of the outcome are used as inputs for the models to forecast dependent variables (Ali et al., 2009). By incorporating influences of external factors that are highly correlated with the demand, causal forecasting can account for changes in the environment or market conditions that affect the outcome. The statistical risk, which refers to the performance of a forecasting model in terms of its accuracy in forecasting future value values based on historical data, and causal risk, which involves the performance of the model in forecasting the effect of changes in the independent variables on the dependent variable, of a model can differ significantly even when assuming causal sufficiency. Causal implications of causal forecasting methods in literature therefore remains largely unexplored (Vankadara et al., 2022) and applications of causal forecasting techniques are very limited (Axsäter, 2015). Causal forecasting is considered complex as it requires extensive theory, domain knowledge, experimental data, and careful consideration when formulating causal

models (Green & Armstrong, 2015). Incorporating explanatory variables into the forecast proved effective in the M5 competition Makridakis et al. (2022). However, rather than traditional regression models, ML models were used to capture complex relationships between variables.

3.1.4 Machine Learning Forecasting Models

In contradiction to statistical models, ML models explicitly evaluate the spectrum or the covariance of the stochastic process, maintaining its empirical structure (Gautam & Singh, 2020). These models attained prominence over the last decade as opposed to statistical models (Gautam & Singh, 2020). However, scant and mixed evidence is available their relative performance in terms of accuracy and computational requirements and more comparison is required for meaningful comparison (Makridakis et al., 2018). Cerqueira et al. (2019) confirms this and states that it is advisable to include both TSF types to ensure completeness of the experimental setup. An inherent limitation of ML models relative to traditional statistical ones is that they are unable to generalise from small datasets (Cerqueira et al., 2019). However, in larger time series, ML has proven to be effective for demand forecasting (Caroleo et al., 2024; X. Zhu et al., 2021; Kim et al., 2020). Particularly the inherent sparsity in intermittent series renders the statistical TSF methods impractical to generate accurate forecasts (Karthikeswaren et al., 2021).

Bernard (2021) distinguishes between three ML paradigms: Supervised learning, unsupervised learning, and reinforcement learning. Supervised learning is a subset of ML and learns a mapping between a set of input variables and an output variable and applies this mapping to predict the outputs for unseen data in classification or regression problems (Cunningham et al., 2008). Classification problems require a discrete value for the output variable, while regression problems result in a continuous value. Unsupervised learning predicts the unseen data by exploring underlying patterns without using labelled data (Ghahramani, 2004). Last, reinforcement learning aims to find an optimal sequence of actions which optimises the expected reward by rewarding or penalising desired or undesired behaviours, respectively (Sutton & Barto, 2018). Primarily regressive supervised learning models are applied in literature, where the observed demand represents the input variable and the forecast demand represents the output variable in the context of demand forecasting.

Supervised ML methods contain a large set of algorithms that are continuously improving (Nasteski, 2017). Many supervised ML methods, particularly Artificial Neural Networks (ANNs), have been proposed as an alternative to statistical ones (Makridakis et al., 2018). Despite mixed evidence, Rosienkiewicz (2013) reported dominance of ANN for spare part demand forecasting-characterised by lumpy demand patterns — and Carmo and Rodrigues (2004); Gutierrez et al. (2008) found that ANN models, even under a relatively simple network topology, generally outperform traditional TSF methods in irregular demand processes. However, Bengio et al. (2017) showed that deep learning networks are generally more accurate than shallow neural networks. Liu and Wang (2024) state that deep learning models have shown the ability to improve the accuracy of specifically univariate and multivariate time series forecasts. This is confirmed by numerous studies (Avinash et al., 2024; Makridakis et al., 2023; Maleki et al., 2024; Taib et al., 2025). Deep learning, a subset of ML, uses neurally inspired deep neural networks to model complex patterns and representations in data, enabling computers to perform cognitive tasks such as image recognition, natural language processing, and TSF with high accuracy (Cichy & Kaiser, 2019). Deep neural networks are formally ANNs with more than three layers, i.e., more than one hidden layer (Sze et al., 2017). Figure 3.1 depicts a Venn diagram of ML-related concepts and classes (Janiesch et al., 2021).

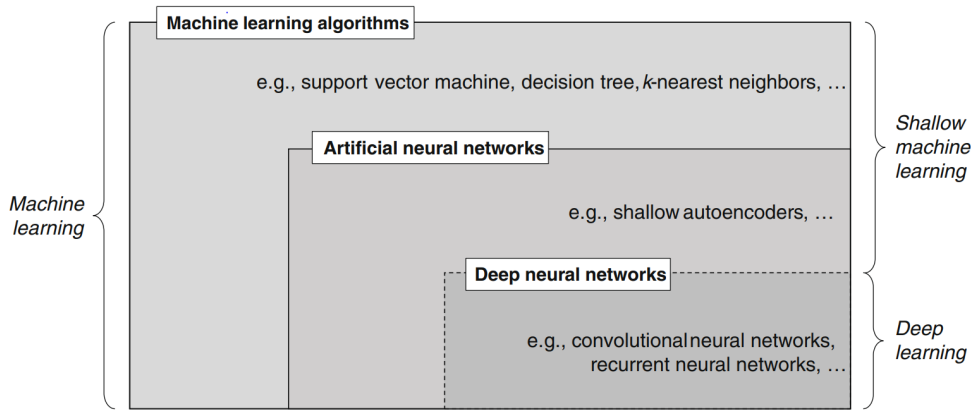
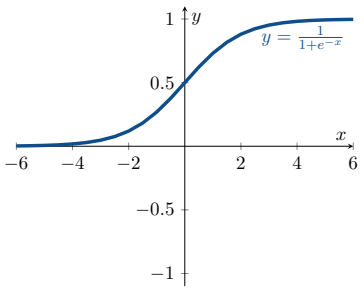


FIGURE 3.1: Venn diagram of ML concepts and classes (Janiesch et al., 2021)

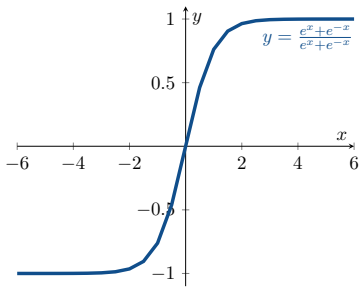
Liu and Wang (2024) differentiates between four deep neural network architectures that are used in TSF: Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), transformers, and Multilayer Perceptrons (MLPs).

Multi-Layer Perceptrons

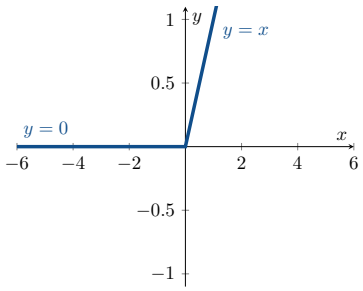
A general architecture of ANNs comprises an input layer, one or more hidden layers, and an output layer each containing multiple neurons. Each neuron in the hidden layers and output layer receives a weighted sum of its input values and performs a functional operation on the weighted input values using a non-linear activation function. The ANN can therefore derive a non-linear relationship between the input layer and output layer (Hoffmann et al., 2022). Commonly used activation functions are depicted in Figure 3.2, including the sigmoid, hyperbolic tangent, Rectified Linear Unit (ReLU), leaky ReLU, and Exponential Linear Unit (ELU) (Sze et al., 2017).



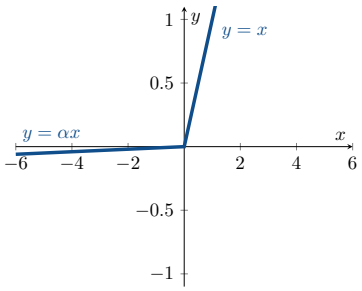
(A) Sigmoid



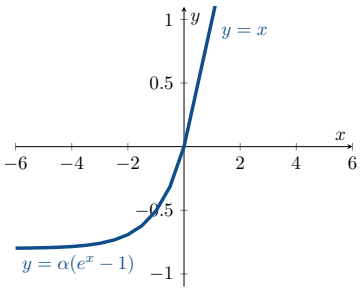
(B) Hyperbolic tangent



(C) ReLU



(D) Leaky ReLU



(E) ELU

FIGURE 3.2: Non-linear activation functions

While many ANN structures exist, the standard model used in TSF is the feedforward neural network or MLP (Muhaimin et al., 2021). In MLPs all computations are performed sequen-

tially using the outputs of the previous layer and are propagated to the next layer as shown in Figure 3.3a. Note that the figure depicts fully connected layer, where all weights are non-zero. These type of ANNs have no memory, so the output for an input is always the same irrespective of the sequence of inputs previously given to the network. Let $f(\cdot)$ denote some non-linear activation function. We denote $x \in \mathbb{R}^m$, $U \in \mathbb{R}^{m \times k}$, and $b^h \in \mathbb{R}^k$ as the input activations, weights connecting the input layer and hidden layer, and the bias vector of the hidden layer, respectively if we assume m input neurons and k neurons in the hidden layer (Sze et al., 2017). Vector $h \in \mathbb{R}$ contains the activation of the neurons in the hidden layer and is computed as (Sze et al., 2017):

$$h = f(U \cdot x + b^h) \tag{3.11}$$

For n neurons in the output layer with activation function $g(\cdot)$, weights connecting the hidden layer and output layer $W \in \mathbb{R}^{k \times n}$, and bias vector of the output neurons $b^o \in \mathbb{R}^n$, we obtain the output vector $y \in \mathbb{R}^n$:

$$y = g(W \cdot h + b^o) \tag{3.12}$$

A limitation of MLPs is that they do not fully exploit the underlying structure often present in the data in applications such as computer vision, natural language processing and forecasting Benidis et al. (2022). Furthermore, the MLP architecture requires a fixed input and output size, as the number of neurons in the input and output layers must be predefined. This makes MLPs impractical for forecasting tasks, which typically require varying input and output sizes Benidis et al. (2022). Next, we describe the more complex RNN, CNN, and transformer, which use basic MLPs as an integral part of their architecture.

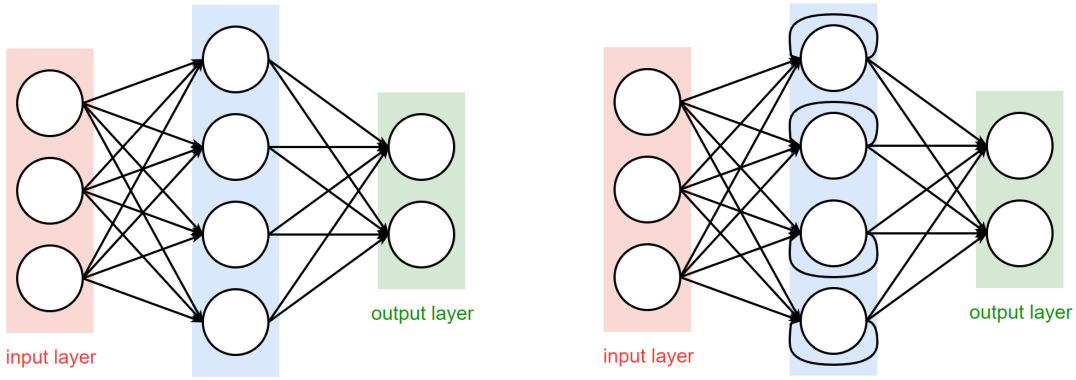
Recurrent Neural Networks

By analogy, RNNs extend the standard MLP to include memory capabilities to allow long-term dependencies to affect the output. It utilises the output of stage $t-1$ as information for state t in the hidden layer as depicted in Figure 3.3b. RNNs can be applied to a variety of temporal processes and are therefore frequently applied to explore time series (Shafi et al., 2023). As in the MLP, the hidden layer and output layer comprise the activation function. With hidden-to-hidden recurrent connections parameterised by weight matrix V, RNN can be mathematically represented as (Shafi et al., 2023):

$$h_t = f(V \cdot h_{t-1} + U \cdot x_t + b^h) \tag{3.13}$$

$$y_t = g(W \cdot h_t + b^o) \tag{3.14}$$

RNNs are powerful for modelling sequential data, but they suffer from potential information loss or vanishing gradient problems (Zhang et al., 2023). Specifically with long sequences, relevant information from earlier time steps in a sequence may get diluted over time. Vanishing gradient problems are encountered while training ANNs during backpropagation. In gradient-based learning methods for training, weights are updated proportional to the gradient value after each iteration. This gradient value can become extremely small, preventing the network from updating its weights. This may stop the ANN from training (Basodi et al., 2020).



(A) Multilayer Perceptrons (MLPs) (B) Recurrent Neural Networks (RNNs)

FIGURE 3.3: Illustration of the MLP and RNN architectures

Long short-term memory Schmidhuber and Hochreiter (1997) proposed the LSTM model to overcome the potential information loss or vanishing gradient problems inherent to RNNs. LSTM is a kind of RNN capable of effectively modelling long-term dependencies between different time steps in sequence by introducing a gating mechanism (including one input gate, one forget gate, and one output gate) to control the information stored in the cell state, i.e., the memory of the architecture, as depicted in Figure 3.4 (Kiefer et al., 2021; Liu & Wang, 2024).

At any time t, the LSTM architecture receives input vector $X_t \in \mathbb{R}^{n \times d}$, previous hidden state vector $H_{t-1} \in \mathbb{R}^{n \times h}$, and previous cell state vector $C_{t-1} \in \mathbb{R}^{n \times h}$, where we denote n as the number of samples in a batch, h as the number of cells in the hidden layer, and d as the number of inputs (Zhang et al., 2023). The forget gate helps keep the most relevant information and forget irrelevant information from X_t and H_{t-1} with weights W_{xf} and W_{hf}, respectively by using a sigmoid activation function σ. The forget gate outputs the following (Zhang et al., 2023):

$$F_t = \sigma(X_t W_{xf} + H_{t-1} W_{hf} + b_f) \tag{3.15}$$

The input gate I_t quantifies the importance of the candidate cell by using a sigmoid activation function as shown in Equation (3.16). Candidate cell state $\tilde{C}_t$ is a proposed update to the cell based on the current inputs X_T and the previous hidden states H_{t-1}. It is computed by applying the hyperbolic tangent (tanh) activation function to a weighted combination of inputs as shown in Equation (3.17) (Zhang et al., 2023).

$$I_t = \sigma(X_t W_{xf} + H_{t-1} W_{hf} + b_f) \tag{3.16}$$

$$\tilde{C}_t = \tanh(X_t W_{xc} + Ht - 1 W_{hc} + b_c) \tag{3.17}$$

The output gate O_t combines the current inputs X_t and the previous hidden states H_{t-1} using the sigmoid function to generate the output vectors for the current period and to determine the next hidden states H_t as shown in Equation (3.18). The current output O_t and the long-term memory C_t are used to update the hidden states as shown in Equation (3.19), where $\odot$ refers to the element-wise Hadamard product operator (Zhang et al., 2023).

$$O_t = \sigma(X_t W_{xo} + Ht - 1 W_{ho} + b_o) \tag{3.18}$$

$$H_t = O_t \odot \tanh(C_t) \tag{3.19}$$

The cell state C_t determines what information to carry over to the next period by adding the Hadamard product of the forget gate F_t and the previous cell state C_{t-1} and the Hadamard product of the input gate I_t and the candidate $\tilde{C}_t$ as shown in Equation (3.20).

$$C_t = F_t \odot C_{t-1} + I_t \odot \tilde{C}_t \tag{3.20}$$

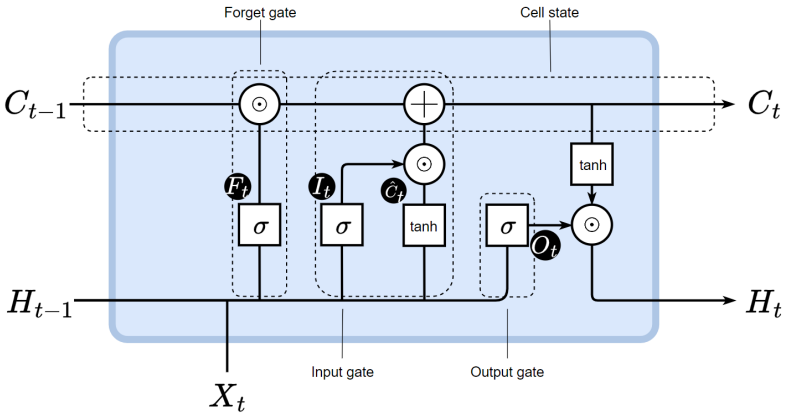


FIGURE 3.4: Illustration of the LSTM architecture

Convolutional Neural Networks

The CNN is a feedforward ANN capable of extracting features from data with convolution structures (e.g., images, videos, audio, time series) using convolution and pooling operations (Li et al., 2021; Liu & Wang, 2024). This type of ANN is commonly applied in classification problems (Borovykh et al., 2017). CNNs replace the weighted sums from the ANN with convolutions of local connections for feature extraction in the convolution and pooling layers and uses a fully connected layer to produce the output (Nguyen et al., 2019). A group of local connections can share the same weight, effectively reducing the parameters (Li et al., 2021). The advantage of using CNN-based models for TSF tasks is in feature extraction, which becomes particularly evident in multivariate cases (Liu & Wang, 2024). The general CNN architecture comprises an input layer, convolution layers, pooling layers, fully connected layers, and an output layer as shown in Figure 3.5, where each layer generates a successively higher-level abstraction of the feature map (Sze et al., 2017).

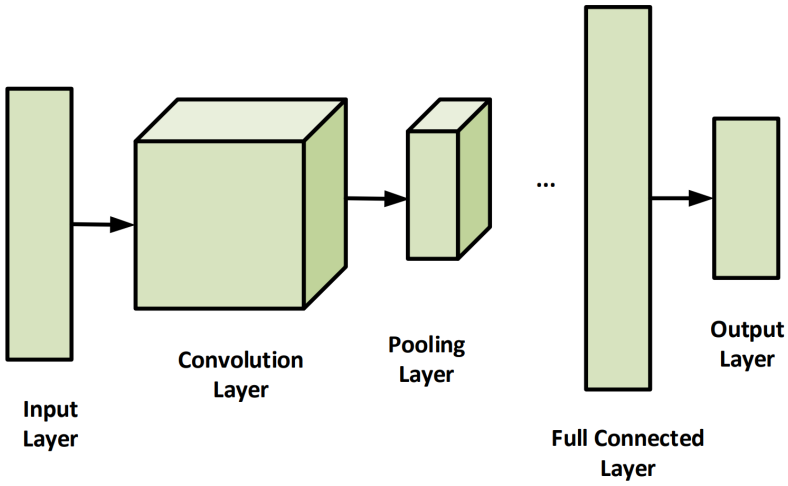


FIGURE 3.5: Illustration of the CNN architecture (Nguyen et al., 2019)

Convolution layer The convolution layer, as displayed in Figure 3.6, comprises a set of convolution kernels, or filters, which slide over a predefined fixed-size window on the feature map passed by the input layer (e.g., a time series or an image) to extract various adjacent feature tiles sequentially (Cong & Zhou, 2023). A weighted sum using element-wise Hadamard multiplication is then computed of each feature tile using the same set of weights, i.e., a convolution kernel, for every channel (Sze et al., 2017). This operation, called a convolution, is used to generate a higher-level abstraction feature map. Non-linear activation functions are then typically applied after each convolution layer (Sze et al., 2017) and feature maps from each filter are reassembled to acquire a new tensor. Mathematically, we formulate the convolution operation including activation function as (Cong & Zhou, 2023):

$$x_j^l = f^l(u_j^l) \tag{3.21}$$

$$u_j^l = \sum_{i \in M_j} x_i^{l-1} * k_{ij}^l + b_j^l \tag{3.22}$$

Here, x_j^l represents the output feature map of channel j in convolution layer l, $f^l(\cdot)$ denotes some activation function of convolution layer l, and u_j^l is the net activation of channel j in convolution layer l. Additionally, let M_j denote the subset of feature tiles sampled by the sliding window of channel j. Last, k_{ij}^l and b_j^l of convolution layer l represent the convolution kernel matrix from input feature map i to output feature map j and the bias of the output feature map j, respectively (Cong & Zhou, 2023).

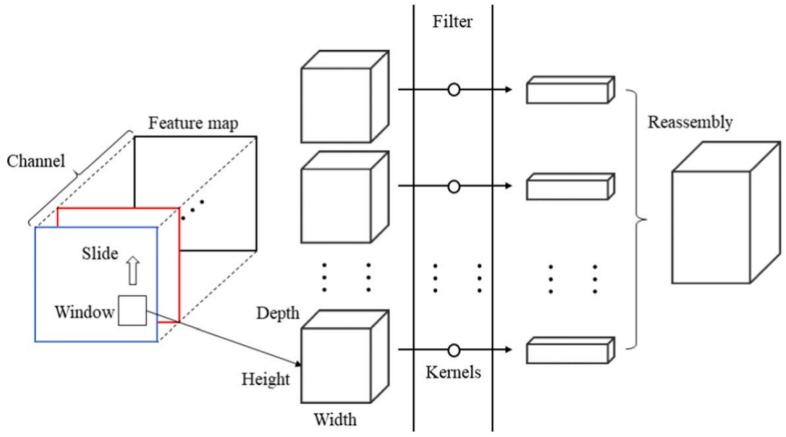
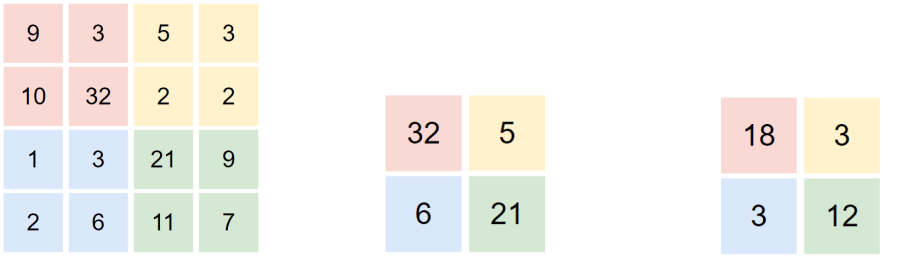


FIGURE 3.6: Working diagram of the convolution layer (Cong & Zhou, 2023)

Pooling layer The pooling or downsampling layer reduces the dimensionality of a feature map by acting as a subsample layer. This layer downsamples the input feature maps of each channel independently by implementing max or average pooling, reducing the network parameters and ensuring a more robust network to small variations (Nguyen et al., 2019). The stride of the pooling refers to how much the pooling window shifts across the input feature map. Pooling typically occurs on non-overlapping receptive fields, i.e., a stride equal to the size of the receptive field (Sze et al., 2017). Figure 3.7 displays a max and average pooling example with a two-by-two receptive field and stride two adopted from (Sze et al., 2017).



(A) Feature map (B) Max pooling (C) Average pooling

FIGURE 3.7: Max and average pooling example (Sze et al., 2017)

Fully connected layer The fully connected layer converts the multi-dimensional tensor format into the final output format and resembles the hidden layer of the MLP (Cong & Zhou, 2023). Recall that the hidden layers of MLPs obtain the output by applying an activation function to the weighted sum of the input activations. In univariate TSF, the fully connected layer is typically used to generate one-dimensional regression information (Mehtab & Sen, 2022).

Transformers

Transformers, like many other ANNs for sequence transduction, have an encoder-decoder structure as depicted in Figure 3.8. This means that an encoder maps an input sequence of continuous or symbol representations to a sequence of continuous representations. Given these encoder output representations, a decoder then generates an output sequence (Vaswani et al., 2017). The encoder and decoder of transformer-based models are composed of N ($N = 6$ in the original Transformer) identical layers which consist of multi-head self-attention and position-wise fully connected feedforward ANN sub-layers. Each decoder layer also inserts a third sub-layer between the self-attention and fully connected layer, which is a multi-head self-attention mechanism applied to the the output of the encoder stack (Vaswani et al., 2017; Wen et al., 2022).

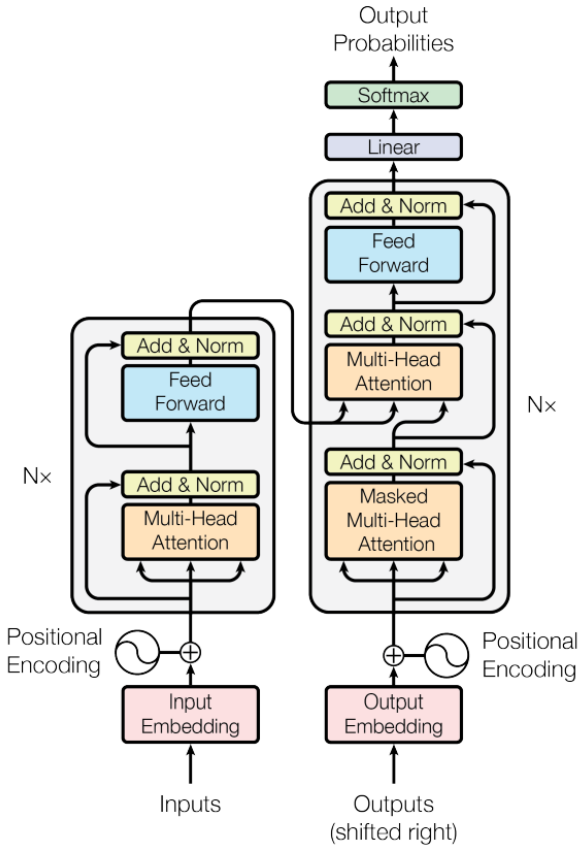


FIGURE 3.8: Illustration of the Transformer architecture (Vaswani et al., 2017)

Self-attention A self-attention mechanism assigns weights to different elements of a sequence when generating an output sequence. It allows input interactions to determine the importance of the information (Zhang et al., 2023). Self-attention uses the vectors query (Q), key (K), and value (V) as input for the softmax function to calculate the importance of the current input relative to other inputs in the previous sequence, which is referred to as the self-attention score. The softmax activation function defined as $\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}}$ is an extension of the sigmoid function for multi-class classification and produces a vector of probabilities. With key dimensionality d_k, we compute the score matrix as (Vaswani et al., 2017; Zhang et al., 2023):

$$A(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{3.23}$$

This scaled dot product represents a single attention function. However, Transformer uses h parallel attention layers with different, learned linear projections $W_i^Q \in \mathbb{R}^{d_{model} \times d_q}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, and $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$ which is referred to as multi-head attention (Vaswani et al., 2017; Q. Zhu et al., 2023). Here, we define d_{model} as the input dimensions. The output of the multi-head attention mechanism is given by the concatenation of each self-attention head $i \in [1, ..., h]$ (Vaswani et al., 2017). Note that we use $W^O \in \mathbb{R}^{hd_v \times d_{model}}$ and $\cdot$ as concatenation operator in Equation (3.25) (Hao et al., 2021).

$$H_i = A(QW_i^Q, KW_i^K, VW_i^V) \tag{3.24}$$

$$\text{MultiH}(Q, K, V) = [H_1, \cdots, H_h]W^O \tag{3.25}$$

Transformers uses multi-head attention in the encoders, where the keys, values, and queries come from the output of the previous layer in the encoder (Vaswani et al., 2017). It is also used

in the decoders to allow each layer in the decoder to attend to information in the sequence up until the current position. Since Transformer processes inputs in parallel, a masked multi-head attention block in the decoder is used which ensures that the autoregressive property is preserved, i.e., future outputs depend only on previous positions in the sequence without accessing future (illegal) information. Masking out illegal connections is done by setting all input values of the softmax activation function corresponding to illegal connections to $-\infty$ (Vaswani et al., 2017). Last, Transformer uses multi-head attention in "encoder-decoder attention" layers. Here, the memory keys and values are delivered by the encoder, while the queries originate from the output of the previous decoder. Multi-head attention is used here to enable each position in the decoder to consider all positions from the entire input sequence (Vaswani et al., 2017).

Feedforward networks The fully connected feedforward networks in each layer are applied to each position of the sequence separately and identically (Vaswani et al., 2017). The inputs X to the neural network are first transformed linearly using weight matrix W_1 and bias vector b_1: $X \rightarrow XW_1 + b_1$. The transformed function is then used as input for the ReLU activation function, which replaces negative values with zero and introduces non-linearity to the model. The activation function output then undergoes a second linear transformation using another weight matrix W_2 and bias vector b_2. We obtain as output (Wen et al., 2022):

$$\text{FFN} = \text{ReLU}(XW_1 + b_1)W_2 + b_2 \tag{3.26}$$

Positional encoding Positional encodings are added to the input embeddings of the encoder and decoder to model the sequence information, since Transformer does not contain recurrence or convolution (Wen et al., 2022). Vaswani et al. (2017) generate a unique vector for each position in the sequence, where each dimension of the positional encoding corresponds to a sinusoid. The even-indexed elements of the positional encoding vector use a sine function, while the odd-indexed elements use a cosine function:

$$PE_{(k,2i)} = sin(\frac{k}{n^{2i/d\text{model}}}) \tag{3.27}$$

$$PE_{(k,2i+1)} = cos(\frac{k}{n^{2i/d\text{model}}}) \tag{3.28}$$

Here, $k \in [0, ..., L-1]$ represents the position of the token in the sequence, and $i \in [0, ..., \frac{d_{\text{model}}}{2}]$ refers to the dimension index of the positional encoding vector. This sinusoidal positional encoding was adopted by Vaswani et al. (2017) with $n = 10,000$ to enable model extrapolation to longer sequences than those encountered during training.

3.2 Forecast Evaluation Metrics

Hyndman (2006) defines four types of forecast evaluation metrics: scale-dependent metrics provide a forecast error on the same scale as the data, percentage error metrics are scale-independent and are therefore used to compare forecasts between different time series, relative-error metrics are scale-independent errors which are divided by the error of some benchmarking method, and scale-free metrics normalise errors to allow comparison across different time series. Lolli et al. (2017) state that a single accuracy measure is generally not sufficiently informative on the different dimensions of the error and suggest using different accuracy measures. Three accuracy measures are therefore proposed in this section.

3.2.1 Scale-Dependent Errors

Scale-dependent metrics are useful when comparing different methods applied to the same time series, but should not be applied, for example, when comparing different methods across time series with different scales (Hyndman & Koehler, 2006). The MSE is one of the most commonly used scale-dependent metrics and it measures the degree of difference between the actual value and forecast value of the model (Liu & Wang, 2024). It is sensitive to outliers as it penalises larger prediction errors more severely since all errors are squared (Chopra & Meindl, 2007). Therefore, (Hyndman, 2006) proposes the Mean Absolute Error (MAE) and we include it to measure the error relative to the observed values and for assessing accuracy on a single series as it is easiest to understand and compute. We denote n as the number of data points available in-sample. MAE is calculated as the arithmetic mean of the absolute errors:

$$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n}|x_t - \hat{x}_t| \tag{3.29}$$

A limitation of the MAE recommended by Hyndman and Koehler (2006) is that it does not show whether the model systematically overpredicts or underpredicts as it regards absolute values. To give an indication of underestimation or overestimation of the forecast, we introduce the Mean Error (ME) to complement the former (Doszyń, 2022):

$$\text{ME} = \frac{1}{n}\sum_{t=1}^{n}(x_t - \hat{x}_t) \tag{3.30}$$

3.2.2 Percentage Errors

Percentage error metrics such as the Mean Absolute Percent Error (MAPE) and symmetric Mean Absolute Percent Error (sMAPE) are scale-independent and can therefore be used to assess the forecast accuracy across multiple time series. However, these metrics have the disadvantage of attaining infinite or undefined values for, e.g., intermittent demand patterns (Hyndman, 2006). Percentage error metrics are therefore not advised if the data contains zeros or small values (Hyndman, 2014). These type of metrics are therefore not used in this study.

3.2.3 Relative Errors

Relative-error metrics divide each error by the error obtained using some benchmark method (e.g., random walk or naive method). An advantage of such metrics is their interpretability, as it measures the improvement of the proposed method relative to the benchmark method (Hyndman & Koehler, 2006). However, relative-error metrics are not used as they may become infeasible as it would involve division by zero if the forecast errors of the benchmark method are zero (Hyndman, 2006).

3.2.4 Scale-free Errors

Hyndman and Koehler (2006) argue that other studies regarding forecast error metrics overlook some fundamental problems and propose the scale-free error metric Mean Absolute Scaled Error (MASE). MASE scales the forecast error based on the in-sample MAE from the random walk (naive) method. MASE is scale-independent and can therefore be used to compare methods across multiple time series. The value of MASE is less than one if it is computed for a more accurate forecast than the average one-step random walk method. Conversely, it is greater than one if the proposed method produces less accurate forecasts than the average one-step random walk forecast (Hyndman & Koehler, 2006). We obtain (Makridakis et al., 2020; Zhang et al., 2023):

$$\text{MASE} = \frac{1}{n} \frac{\sum_{t=1}^{n} |x_t - \hat{x}_t|}{\frac{1}{n_1-1} \sum_{t=2}^{n_1} |x_t - x_{t-1}|} \tag{3.31}$$

MASE can be used to compare forecast methods in single series and across multiple series, it cannot give infinite or undefined values, and it is symmetric (Hyndman, 2006). Hyndman and Koehler (2006) state that it should become the standard accuracy metric for comparing forecasts across multiple time series. Hyndman (2006) also suggests the use of MASE for all forecasting situations, methods, and all types of series. In line with Hyndman (2014), we use MASE for forecast accuracy performance comparison.

3.3 Related Work

Intermittent and lumpy demand patterns are very common in real-world datasets, e.g., in heavy machinery, respective spare parts, aviation service parts, electronics, maritime spare parts, automotive spare parts, and retailing (Gutierrez et al., 2008; Kiefer et al., 2021; Willemain et al., 1994). Croston (1972) concluded that conventional exponential smoothing methods were not particularly well suited for intermittent time series forecasting. He was the first to propose a method aiming to overcome the difficulties of intermittent demand by using separate estimates of the non-zero demand size and demand interval between two consecutive non-zero demand events. Empirical studies have shown superior performance of Croston's proposed method over conventional methods (Levén & Segerstedt, 2004; Syntetos & Boylan, 2001; Willemain et al., 1994; Zhang et al., 2023). Syntetos and Boylan (2001) concluded that Croston's method is biased and presented SBA as a corrected version to overcome the forecast bias and increase the forecasting performance. Gutierrez et al. (2008) and Şahin et al. (2013) provide empirical evidence of SBA's superior forecasting performance over Croston's method and SES using real-world datasets of lumpy spare part demand of an aircraft maintenance, repair and overhaul company and an electronics distributor, respectively.

Fattah et al. (2018) successfully apply ARIMA to forecast future erratic demand in a food manufacturing company. Luochen and Hasachoo (2021) compare various statistical models for forecasting irregular demand in pharmacy operations and find that Croston's corrected SBA method performs best on a erratic and lumpy datasets.

Gutierrez et al. (2008) were, to the best of our knowledge, the first to apply ANN modelling to lumpy demand forecasting. The study applied a simple three-layered MLP with two input nodes: (1) the demand at the end of the immediately preceding period and (2) the number of periods separating the last two non-zero demand transaction at the end of the immediately preceding period. The MLP architecture trained on all 24 univariate time series together by the backpropagation algorithm also comprises three neurons in a single hidden layer and an output node representing the one-step-ahead predicted demand value. Gutierrez et al. (2008) concluded that ANN models are generally superior to traditional TSF models in forecasting lumpy demand.

While Croston's method significantly outperforms the three-layered MLP in a multi-step ahead forecasting study by Kiefer et al. (2021), other studies by Babai et al. (2020), Hoffmann et al. (2022), Lolli et al. (2017), Mukhopadhyay et al. (2012), and Şahin et al. (2013) report superior results for erratic and lumpy one-step ahead demand forecasting using a simple feedforward MLP using backpropagation, based on the study by Gutierrez et al. (2008) compared to Croston-based methods and a RNN in the latter study. Amin-Naseri and Tabar (2008) report good results of a simple RNN structure with eight input variables using the backpropagation training algorithm for forecasting one-step ahead lumpy spare part demand, outperforming the basic MLP structure.

Abbasimehr et al. (2020) and Kiefer et al. (2021) report superior forecasting performance of LSTM in their respective studies. Kiefer et al. (2021) found that a single hidden layer LSTM model outperforms a LSTM with two hidden layers for forecasting intermittent and lumpy demand across all types, while Abbasimehr et al. (2020) successfully uses a two-hidden-layer LSTM for forecasting non-linear, non-stationary demand after grid search hyperparameter optimization.

Mirshahi et al. (2024), in their 2-step CNN-based approach, applied two separate CNNs to forecast real-world intermittent time series demand for optimised supply chain management. An initial CNN is used as a binary classifier to predict demand occurrence, while a second CNN estimates the demand size. Although limited empirical evidence was found in the erratic and lumpy TSF domain, Bai et al. (2018) state that CNN-based models should be included in sequence modelling studies.

Zhang et al. (2023) was, to the best of our knowledge, the first to empirically test the effectiveness of Transformer in forecasting of intermittent demand of an airline spare parts provider. The results showed that Transformer outperforms Croston's method, SBA, and state-of-the-art ANN architectures including MLP, RNN, and LSTM. Helgesson and Laszlo (2023) successfully applied Temporal Fusion Transformer, a transformer-based model introduced by Lim et al. (2021), for multi-horizon forecasting using Cross-Learning (CL) on clusters of homogeneous time series.

TABLE 3.3: Overview of Erratic and Lumpy Demand Forecasting Literature

Paper	Time series[a] U	M	Frequency[b]	Normalisation	CL	Model	Horizon[c]	Forecast measure(s)
Abbasimehr et al. (2020)	✓		M	Min-max		LSTM	26	sMAPE, RMSE
Amin-Naseri and Tabar (2008)		✓	M	Min-max		RNN	1	MAPE, MASE, PB
Babai et al. (2020)	✓		M			MLP	1	sME, MSE, MASE
Fattah et al. (2018)	✓		M			ARIMA	10	ME
Gutierrez et al. (2008)	✓		D			MLP	1	MAPE, PB, RGRMSE
Helgesson and Laszlo (2023)		✓	D	Z-score	✓	TFT	48	ME, MAE, PB
Hoffmann et al. (2022)	✓		M			MLP	1	MAPE
Kiefer et al. (2021)	✓		D	Z-score		Croston	28	MASE, SPEC
Lolli et al. (2017)	✓		W			MLP	1, 3, 5	MAPE, ME/A
Mirshahi et al. (2024)		✓	W	Min-max		CNN	1	MAE, RMSE
Mukhopadhyay et al. (2012)	✓		D			MLP	1,5	MAPE, MdRAE, RGRMSE, PB
Şahin et al. (2013)	✓		M	Min-max		MLP	1	GMAMAD/A
Luochen and Hasachoo (2021)	✓		D			SBA	90	MSE
Zhang et al. (2023)	✓		W	Min-max		Transformer	1	ME, MAE, RMS, MAPE, MASE, PB
THIS PAPER	✓	✓	6M	Min-max			1	ME, MAE, MASE

[a] U=Univariate, M=Multivariate.
[b] t=t-minute interval, H=Hourly, D=Daily, W=Weekly, M=Monthly, 6M=Six-Monthly, Y=Yearly. [c] In same units as frequency.

Table 3.3 summarises related work concentrated on erratic and lumpy demand forecasting for short-to medium horizons using real-world datasets. Despite extensive research on TSF in general, limited literature is available on TSF for erratic and lumpy demand patterns in real-world datasets. We observe that although many complex deep neural network architectures have outperformed statistical models, Croston-based methods occasionally outperform these state-of-the-art approaches. As no single best practice emerges from the literature, we include a diverse set of statistical models and complex ML architectures in our experimental setup to ensure a comprehensive comparative evaluation.

Although some evidence suggests CL to improve forecast accuracy (Montero-Manso & Hyndman, 2021), limited empirical evidence is found in related literature. Based on Table 3.3, min-max normalisation is the most commonly applied data pre-processing technique before feeding the

data into the ML models.

3.4 Findings and Implications

This chapter provides the theoretical framework of the research — forming the foundation of Phase 4: Solution Formulation — by detailing best practices in the demand forecasting domain. Moreover, studies with similar characteristics to the presented use-case are reviewed to identify potential solution formulations. Accordingly, research question 4 and 5 are answered.

4. What forecasting models does the literature propose for forecasting demand in the HNL use-case?

An analysis of forecasting techniques is given in Section 3.1. Particularly univariate and multivariate statistical TSF models and deep neural networks were found to be suitable for the HNL use-case. Since we found mixed and scant evidence in literature (see discussion in Section 3.1.4), we include a broad and diverse set of statistical and deep learning models to ensure a meaningful comparative analysis as proposed by Cerqueira et al. (2019). Section 3.3 similarly highlights the absence of a clear consensus, further supporting the inclusion of a broad range of models for a meaningful evaluation. We particularly concentrated on two exponential smoothing methods, Croston-based methods, ARIMA, and three classes of deep neural networks — RNNs, CNNs, and transformers – due to their prominence in TSF tasks relative to shallow ML models.

5. What metrics are proposed in literature to ensure robust forecasting model evaluation?

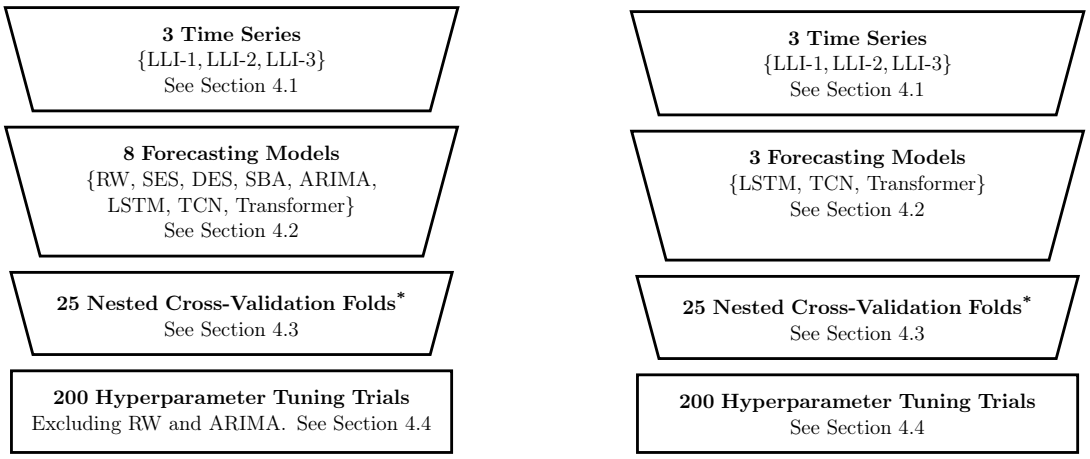
In Section 3.2, we evaluated forecast evaluation metrics. It is advised in literature to apply multiple metrics, as a single accuracy measure is not sufficiently informative. Key findings from this section include:

- The ME is proposed to give an indication of underestimation or overestimation of the forecast.
- The former should be complemented by the MAE to give an indication of the average magnitude of forecast errors regardless of the direction.
- Moreover, the scale-independent error metric MASE is proposed as metric to compare accuracy across multiple time series.

Chapter 4

Experimental Setup

Given the established theoretical framework in Chapter 3, this chapter formulates the experimental setup that guides the solution selection for the presented use-case. As outlined in Section 1.4, research questions 6 and 7 — related to Phase 4: Solution Formulation — are addressed. A high-level overview of the experimental setup is provided in Figure 4.1, which visually summarises the flow from time series inputs through model application, validation, and hyperparameter tuning. Section 4.1 outlines the data preprocessing steps adopted in this study. Moreover, Section 4.2 details the implemented forecasting models, while Section 4.3 outlines the procedures used to evaluate their performance. Last, hyperparameter tuning is addressed in Section 4.4.



(A) Univariate TSF (B) Multivariate TSF

FIGURE 4.1: Experimental setup summary for univariate (a) and multivariate (b) TSF model validation. *LLI-1 and LLI-3 used 25 folds; LLI-2 used 26 folds.

4.1 Data Collection and Preprocessing

This section outlines the data collection and preprocessing steps for both univariate and multivariate forecasts. It covers the demand data (target variable) in Section 4.1.1, and the covariate data used in multivariate forecasting in Section 4.1.2. In line with the findings from Chapter 3, we normalised all time series using the min-max normalisation algorithm, given by:

$$x_{\text{normalised}} = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{4.1}$$

4.1.1 Demand Data

We use a demand dataset comprising all part issues recorded from July 2003 to November 2024, including, e.g., issue date, part code, quantity, and transaction type. The given dataset of historical part issues includes part codes that replaced earlier part codes associated with the same unique SKU. Consequently, all part codes were mapped according to their description column in the dataset, which identifies potential replacements, and the original part codes for each SKU were used. Next, all part issues are aggregated into time buckets of six months and we impute missing values with zero. The data is filtered based on the selected LLIs, HNL, and part issues for the New Build business line. We remove periods of zero demand preceding the first non-zero value, as these represent times when the respective items were not yet in use. Including them would introduce an unrepresentative demand pattern, potentially distorting model training and evaluation. This results in 39 data points for LLI-1 and LLI-3, and 40 for LLI-2.

4.1.2 Covariate Data

This section details the covariate data used as input for the multivariate forecasting models, including temporal and project-related data within the ETO environment of HNL. Since the Darts implementations of LSTM, Temporal Convolutional Network (TCN), and Transformer only support past covariates (i.e., covariates known only into the past), all covariate data is implemented accordingly.

Temporal Encodings

Temporal encodings are added to provide the multivariate models with contextual time information that potentially helps capture demand patterns. Specifically, we include three features: the calendar year, a binary half-year indicator (0 for January–June time buckets, 1 for July–December time buckets), and the relative position of each period in the time series — all normalised using min-max normalisation. A visual representation of the temporal encodings can be found in Appendix D.

Project-related Data

Project-related covariates are included to reflect the project-based operational context of HNL driving demand. We first define a classification scheme to group projects by type based on the equipment type produced (see Appendix E). Subsequently, we collect data on all historically contracted projects, including their start dates, assigned project class, and the estimated number of sheaves required during the concept design phase. Since the three forecasted bearing LLIs are primarily used within sheave assemblies, we assume that the estimated number of sheaves may be indicative of future bearing demand. The data is then aggregated into six-month time buckets, yielding two covariates per period per project class: the number of projects and the total number of sheaves summed over all signed projects within that period. Last, we normalise all project-related data using min-max normalisation.

4.2 Model Specification and Development

This section outlines the models evaluated in this study to address the HNL forecasting task. A naive Random Walk (RW) model is included as a simple, interpretable benchmark to contextualize the performance of more complex statistical and ML models and to assess the inherent predictability of the time series. Following the findings from Chapter 3 — in line with the "no free lunch" theorem, which states that no algorithm is universally superior in all scenarios (Wolpert & Macready, 1997) — multiple statistical and ML models are evaluated for the HNL use-case. We develop these models using the Darts API, an open-source ML Python library for

end-to-end TSF tasks (Herzen et al., 2022). Refer to Section 3.1 for an overview of model architectures. Moreover, for a description of the hyperparameter tuning methods and configuration spaces, refer to Section 4.4.

4.2.1 Benchmark Model

Beck et al. (2025) highlights the importance of including a simple benchmark model for comparison against more complex fitted models to gain insights into the inherent predictability of highly volatile time series. In this study, we adopt the naive RW model, which simply sets all forecasts to be the value of the previous observation (Hyndman & Athanasopoulos, 2018). The benchmark model involves no parameters and is applied exclusively in the univariate forecasting case.

4.2.2 Statistical Models

This study includes two exponential smoothing methods, the Croston-based SBA, and ARIMA, which are traditionally used for univariate forecasting and are applied exclusively to univariate time series in this study.

Exponential Smoothing

Chapter 2 indicates that two selected time series exhibit significant trend without seasonality, whereas the third series shows neither a significant trend nor seasonality. We therefore use conventional SES and DES models, which assume constant and linear trend demand models, respectively (see Section 2.4.2 for these demand models). We use the `ExponentialSmoothing` class from the Darts library, which is essentially a wrapper around Statsmodels' Holt-Winters' exponential smoothing implementation. Note that the SES and DES models are obtained by configuring the trend and seasonal components of the `ExponentialSmoothing` class accordingly. Optimal values for the smoothing factors α and β are determined using hyperparameter optimisation. Additionally, hyperparameter optimisation determines whether a damped trend component should be included, where the damping factor ϕ is automatically induced during model fitting.

Syntetos-Boylan Approximation

We implement the Croston-based SBA in our comparative evaluation, as it corrects the bias of Croston's method and has proven to be a superior statistical method in forecasting sporadic demand. The model is custom-developed in Python for the univariate forecasting case, with its smoothing factor α determined through hyperparameter optimisation.

Autoregressive Integrated Moving Average

For the widely applied ARIMA in TSF, we automate the configuration of optimal p, d, and q values using `AutoARIMA` in Darts, which is based on the `AutoARIMA` class from the Statsforecast package. We adopt minimisation of Akaike's Information Criterion corrected (AICc), which is the default behaviour in the model class. Moreover, Hyndman and Khandakar (2008) use the AICc in their algorithm for automatic ARIMA modelling, which was also proposed by Hyndman and Athanasopoulos (2018).

4.2.3 Machine Learning Models

This study also includes three ML models — LSTM, TCN, and Transformer — which are applied to both univariate and multivariate forecasting tasks. We use the default Adam optimiser for training all ML models, as proposed by (Kingma & Ba, 2014).

Long Short-Term Memory

To overcome the potential information loss or vanishing gradient problems inherent to RNNs (see Section 3.1.4), we include the RNN-based architecture LSTM. The model is implemented using Darts' `RNNModel` class with the `model` parameter set to `"LSTM"`, which is built on the PyTorch library. For multivariate forecasting with past covariates, we use Darts' `BlockRNNModel` class configured as an LSTM, enabling the incorporation of additional input series beyond the target variable. Obtaining good performance with LSTM networks is not a simple task, as it involves the optimisation of multiple hyperparameters (Reimers & Gurevych, 2017). A detailed list of hyperparameters tuned — based on the optimised approaches by Abbasimehr et al. (2020) and Reimers and Gurevych (2017) — is given in Section 4.4.

Temporal Convolutional Network

TCN is a CNN-based architecture adapted to the TSF domain. It uses convolutions in the temporal dimension, enabling the architecture to automatically learn temporal and spectral features without information "leakage" from future to past (Bai et al., 2018; Pelletier et al., 2019). The model is implemented using the `TCNModel` class from the Darts library, which use dilated convolutions that enable an exponentially large receptive field by progressively increasing the spacing between kernel elements at each layer (Bai et al., 2018). The dilated convolutions enable the model to capture long-term dependencies. As described in Section 4.4, 10 hyperparameters of the TCN architecture are optimised using hyperparameter tuning.

Transformer

Despite the introduction of a gating mechanism, LSTM may have difficulties interpreting long input sequences because of its sequential data processing nature (Zhang et al., 2023). Besides TCN, we therefore also apply the state-of-the-art Transformer, a deep learning model introduced by Vaswani et al. (2017). It is implemented using Darts' `TransformerModel`, which is based on PyTorch's `Transformer` class. In line with Equation (3.26) and the original formulation by Vaswani et al. (2017), we use `ReLU` as the activation function within the feedforward layers of the Transformer architecture. We tune the hyperparameters listed in Section 4.4 for each time series and for both univariate and multivariate TSF tasks.

4.3 Validation Procedure and Model Evaluation

This section outlines the model selection and model evaluation procedures adopted in the experimental setup of this study. Section 4.3.1 presents the validation procedure adopted to obtain a robust estimate of each model's forecast accuracy, while Section 4.3.2 details the evaluation metrics that are applied to calculate the forecast errors.

4.3.1 Validation Procedure

To validate the forecasting models, we employ cross validation — a statistical method for evaluating and comparing learning algorithms by dividing the data into separate training and testing sets (Refaeilzadeh et al., 2009). However, traditional cross-validation with random splits becomes problematic for TSF tasks due to the temporal dependence between observations, i.e., no independent and identically distributed (i.i.d.) data. Cross-validation with random splits that do not respect the temporal ordering of the series may cause information "leakage" from future to past, leading to over-optimistic performance estimates.

As "a nested cross-validation procedure provides an almost unbiased estimate of the true error" (Varma & Simon, 2006), we use nested cross-validation to estimate the prediction errors of the forecasts. This validation method uses an inner cross-validation loop — which partitions

the outer training set into an inner training set and a fixed-size validation set — to perform hyperparameter tuning, while an outer cross-validation is used to compute an estimate of the error on unseen, out-of-sample data using the best-found hyperparameters for each fold (Varma & Simon, 2006). To respect the temporal dependence in the data, we use rolling-origin-recalibration (Bergmeir & Benítez, 2012) — also referred to as rolling-origin evaluation (Tashman, 2000) — in the outer loop of the nested cross-validation procedure. Forecasts for a fixed horizon are performed by successively moving observations from the out-of-sample test set to the in-sample training set, and changing the forecast origin accordingly (Bergmeir & Benítez, 2012). The model is recalibrated for each forecast to include all available data in the training set, and the evaluation metrics are averaged across all outer loops to obtain a robust measure of the forecast accuracy. This validation approach, illustrated in Figure 4.2, ensures optimal use of all available data while overcoming the main limitation of time series hold-out cross-validation — also referred to as fixed-origin evaluation in literature — where a single arbitrary split is made in the time series, and characteristics of the selected forecast origin might heavily influence evaluation results (Bergmeir & Benítez, 2012).

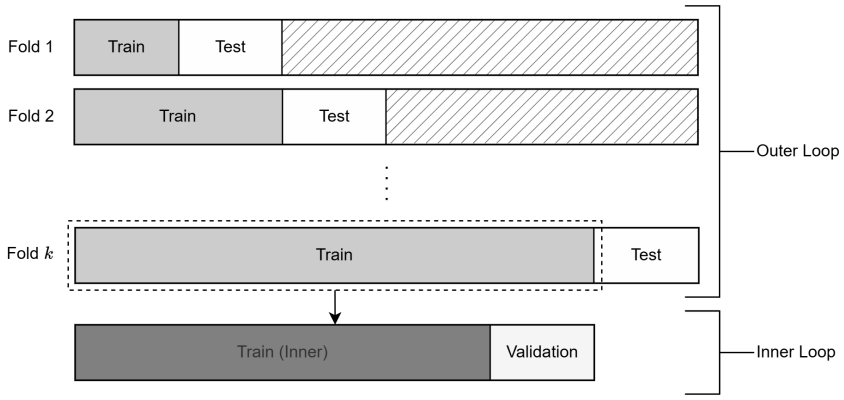


FIGURE 4.2: Illustration of the nested cross-validation procedure (Paik et al., 2023)

The Darts API requires a minimum training set size of 10 data points. Accordingly, the initial inner training set in the first fold consists of 10 observations. To reflect the average project duration of approximately two years, we fix the validation set to four observations (i.e., two years), ensuring that hyperparameters are tuned over the full demand cycle of typical projects. This results in an initial outer training set size of 14. Each test set comprises one observation, as we aim to evaluate one-step-ahead forecast performance on unseen data. With 39 data points for LLI-1 and LLI-3 and 40 data points for LLI-2, this results in 25 validation folds for LLI-1 and LLI-3 and 26 validation folds for LLI-3 as depicted in Figure 4.1.

4.3.2 Evaluation Metrics

In line with the findings from Chapter 3, we apply the ME, MAE, and MASE to evaluate the accuracy of the forecasts. The ME metric gives an indication of the underestimation or overestimation of the forecast, while the MAE provides an indication of the average magnitude of the forecast errors. Last, MASE is applied to enable comparison across time series. This metric is scale-independent, it cannot give infinite or undefined values, and it is symmetric. Additionally, it was used during the M4 competition (Makridakis et al., 2020). The evaluation metrics are calculated within each fold, and we obtain a final estimate of the forecast accuracy by averaging the results across all folds. Moreover, the standard deviations across folds are calculated to assess the consistency and robustness of the models. We use the L1 loss function in the training procedure of our models, i.e., we minimise the MAE.

4.4 Hyperparameter Tuning

As an integral part of the nested cross-validation procedure, we tune each forecasting model's hyperparameters within the inner loop. In each fold, the model is trained on the inner training set, and the loss function is optimised on the validation set. Formally, hyperparameter optimisation in a minimisation problem is defined as follows (Watanabe, 2023):

$$x_{\text{opt}} \in \arg\min_{x \in \mathcal{X}} f(x) \tag{4.2}$$

Here, x_{opt} denotes the optimal hyperparameter configuration, x a candidate hyperparameter configuration, $\mathcal{X}$ the configuration space comprising all possible hyperparameter inputs, and $f(x)$ the objective (or loss) function to be minimised. The optimisation algorithm adopted in this research is the Tree-Structured Parzen Estimator (TPE), which is detailed in Section 4.4.1. Moreover, the configuration spaces of the forecasting models are assessed in Section 4.4.2.

4.4.1 Optimisation Algorithm

We adopt the optimised TPE implementation from Watanabe (2023) using the `CustomizableTPESampler` class, a Bayesian optimisation method widely used in recent hyperparameter tuning frameworks. Bayesian optimisation uses an acquisition function to iteratively search for x_{opt}, balancing the algorithm's degree of exploration and exploitation (Watanabe, 2023).

The TPE algorithm — detailed in Algorithm 1 (Watanabe, 2023) — initially evaluates N_{init} random samples, and stores the initial observations in a set $\mathcal{D}$. Then, using quantile threshold γ defined by Γ, we partition past observations into a better subset $\mathcal{D}^{(l)}$ and a worse subset $\mathcal{D}^{(g)}$ to build two Probability Density Functions (PDFs). TPE then samples N_s candidate hyperparameter configurations from the better group, and determines the configuration with the best acquisition function value (Watanabe, 2023). This configuration is then evaluated against the objective function and added to observations $\mathcal{D}$. This process is repeated until a stopping criterion is reached. In the remainder of this section, we detail the TPE used in this study. For a comprehensive overview of alternative TPE parameters refer to Watanabe (2023).

Algorithm 1: Tree-structured Parzen Estimator (TPE)

Data: Initial parameters: N_{init} (initial samples), N_s (candidate samples), Γ (quantile function), W (weight function), k (kernel function), B (bandwidth function)
Result: Best configuration found
Initialise: $\mathcal{D} \leftarrow \varnothing$

for $n = 1$ **to** N_{init} **do** ▷ Initialisation
  $x_n \leftarrow$ `RandomSample()`
  $y_n \leftarrow f(x_n) + \epsilon_n$ ▷ Evaluate objective function
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(x_n, y_n)\}$
while *NOT stopping_criteria* **do** ▷ Stopping criteria
  Compute $\gamma \leftarrow \Gamma(N)$ with $N := |\mathcal{D}|$ ▷ Splitting algorithm
  Split $\mathcal{D}$ into $\mathcal{D}^{(l)}$ and $\mathcal{D}^{(g)}$
  Compute $\{w_n\}_{n=0}^{N+1} \leftarrow W(\mathcal{D})$ ▷ Weighting algorithm
  Compute $b^{(l)} \leftarrow B(\mathcal{D}^{(l)})$, $b^{(g)} \leftarrow B(\mathcal{D}^{(g)})$ ▷ Bandwidths
  Build $p(x|\mathcal{D}^{(l)})$, $p(x|\mathcal{D}^{(g)})$ using Eq. (5) ▷ Kernel function
  Sample $\mathcal{S} := \{x_s\}_{s=1}^{N_s} \sim p(x|\mathcal{D}^{(l)})$
  Pick $x_{N+1} := \arg\max_{x \in \mathcal{S}} r(x|\mathcal{D})$ ▷ Acquisition function
  $y_{N+1} := f(x_{N+1}) + \epsilon_{N+1}$ ▷ Evaluate objective function
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(x_{N+1}, y_{N+1})\}$
return Best configuration in $\mathcal{D}$

Stopping Criteria

A drawback of our validation procedure (see Section 4.3.1), particularly in combination with the use of deep learning models, is its computational complexity, as each fold requires complete retraining of the model. To ensure our experiments remain computationally tractable, we adopt a pragmatic approach and use 50 trials (i.e., iterations) as the stopping criterion.

Splitting Algorithm

The TPE algorithm splits observations $\mathcal{D}$ into the better subset $\mathcal{D}^{(l)}$ and the worse subset $\mathcal{D}^{(g)}$ using quantile function Γ. Careful selection of the quantile function is crucial, as it balances between exploration and exploitation. A lower value of γ reduces the cardinality of the better set $\mathcal{D}^{(l)}$, encouraging the algorithm to focus on the most promising areas (i.e., exploitation), whereas a larger value includes more configurations in $\mathcal{D}^{(l)}$, ensuring a broader search (i.e., exploration). We adopt a simple linear quantile function given by:

$$\Gamma(N) = \beta_1 = 0.15 \tag{4.3}$$

This implies $\gamma = 0.15$. Each iteration, we therefore partition the top 15% of $\mathcal{D}$ into $\mathcal{D}^{(l)}$, while the rest is assigned to $\mathcal{D}^{(g)}$. This quantile function was found to generalise the most in the study by (Watanabe, 2023), and is therefore applied in this study.

Weighting Algorithm

The weighting algorithm W in the TPE algorithm assigns weights to each observation in D for the PDFs. In line with the findings of Watanabe (2023), we use the expected improvement weighting algorithm by setting the weight strategy to `EI`. The advantage of this algorithm, compared to others, is its ability to consider the ranking of observations in the better subset $\mathcal{D}^{(l)}$ (Watanabe, 2023). Note that the weights sum up to 1 for both subsets.

Bandwidths and Kernel Function

To estimate the PDF, we use a Gaussian kernel (Watanabe, 2023):

$$g(x, x_n|b) := \frac{1}{\sqrt{2\pi b^2}} e^{-\frac{1}{2}(\frac{x-x_n}{b})^2} \tag{4.4}$$

This kernel function is used to build the PDFs for subsets $\mathcal{D}^{(l)}$ and $\mathcal{D}^{(g)}$, with the bandwidth control parameter b determined using the `hyperopt` heuristic. After having determined b, we use so-called magic clipping to optimise the bandwidth. Magic clipping parameters include the minimum bandwidth b_{min} and the exponent α for the magic clipping algorithm. For more details on the applied magic clipping, refer to Watanabe (2023). Watanabe (2023) strongly recommends using a multivariate kernel, which considers the entire vector of hyperparameters at once when determining the probability density, to enhance the performance. Therefore, we set `multivariate` to `True`.

Acquisition Function

Once the PDFs for $\mathcal{D}^{(l)}$ and $\mathcal{D}^{(g)}$ are constructed, TPE uses an acquisition function $r(x|\mathcal{D})$ to determine the potential of a hyperparameter configuration. The acquisition function computes the density ratio between the PDFs of the better subset $\mathcal{D}^{(l)}$ and the worse subset $\mathcal{D}^{(g)}$ (Watanabe, 2023):

$$r(x|\mathcal{D}) := \frac{p(x|\mathcal{D}^{(l)})}{p(x|\mathcal{D}^{(g)})} \tag{4.5}$$

This ratio indicates the probability that a given configuration is included in the better subset $\mathcal{D}^{(l)}$. The TPE algorithm then selects the configuration candidate with the maximum density ratio, balancing exploration and exploitation. High density values for $p(x|\mathcal{D}^{(l)})$ increase the density ratio, encouraging configurations similar to those that previously performed well (exploitation). Conversely, low densities $p(x|\mathcal{D}^{(g)})$ indicate that the configuration is uncommon among poor performing configurations, which also increases the density ratio, thereby promoting less frequently observed configurations (exploration).

4.4.2 Configuration Spaces

To guide the hyperparameter optimisation algorithm TPE, we define the search space for each hyperparameter as the domain of the d^{th} hyperparameter, denoted by $\mathcal{X}_d \subseteq \mathbb{R}$. For a model with D tunable hyperparameters, the hyperparameter configuration space $\mathcal{X}$ is therefore defined as the set of all potential hyperparameter configurations $\mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_D$ (Watanabe, 2023). In this section, we specify the individual search spaces $\mathcal{X}_d$ for each hyperparameter, thereby defining the overall configuration space $\mathcal{X}$ from which the TPE algorithm (see Section 4.4.1) selects candidate configurations to solve the minimisation problem defined in Equation (4.2).

Statistical Models

For the statistical TSF models SES, DES, and SBA a limited number of hyperparameters are involved. Based on findings in Chapter 3, there are no consistent guidelines on the selection of the smoothing factor α, but values between 0 and 0.3 are typical in forecasting literature across the three models. We therefore define the search space of the smoothing factor as $\alpha \in [0, 0.3]$. For the smoothing factor for the trend in DES, relatively low values are recommended in literature (see 3.1.1), and typical values of β are approximately 0.05. To allow exploration during hyperparameter optimisation, we do not restrict the TPE algorithm to only low values for β. Instead, we define the search space of the trend smoothing factor as $\beta \in [0, 0.2]$. Note that we use the `AutoARIMA` class in Darts, which automates the parameter tuning process for ARIMA.

Machine Learning Models

For the batch size hyperparameter, defined as the number of samples used to update the model's weights in a single iteration, researchers generally apply a batch size in the range of 2 to 128 by trial and error (Hwang et al., 2024). Due to the irregular and lumpy demand patterns in this study, we exclude smaller batch sizes to ensure that the model captures broader patterns in the data, rather than overfitting to noise from occasional demand spikes. Therefore, for all deep neural networks, we consider the batch sizes $\{16, 32, 64, 128\}$ within the search space. The dropout rate is used to regularise the neural networks by randomly excluding a fraction of neurons. This overcomes the problem of overfitting by forcing the models to be more robust and not overly dependent on a small subset of neurons. Park and Kwak (2017) empirically test different regularisation methods, and find optimal values for the standard dropout method within the range $[0, 1]$. Therefore, this represents our search space for the dropout rate for all ANNs. The number of epochs is defined as the number of passes the learning algorithm makes over the entire training dataset during the training process. Afaq and Rao (2020) state that there is no universal optimal number of epochs, and the optimal number of epochs differ per dataset. Since we adopt an expanding-window validation scheme, our training data starts relatively small and is complex in later folds. We therefore adopt a broad search space for the number of epochs in each ANN: $[20, 200]$. Given the absence of seasonality in each of the time series and the long lead times inherent to the ETO environment, we adopt a broad lag range of 1 to 13 months to capture medium- to long-term temporal dependencies in the data. Last, we adopt learning rates — which determines the extent to which the model is updated after

each training iteration — in the interval $[1e^{-5}, 1e^{-1}]$ for all ANNs, ensuring the TPE algorithm can comprehensively explore learning rates. Since we use a broad search space for the number of epochs, we adopt a relatively broad search space for the learning rate to enable quicker convergence in configurations with fewer epochs and more stable convergence for configurations with more epochs.

For all model-specific hyperparameters, we take an ad hoc approach by selecting relatively broad search spaces compared to optimal values in literature. Consequently, we enable the TPE algorithm to comprehensively evaluate the configuration space. Table 4.1 details the search space for all tuned hyperparameters of the deep learning models, along with their respective notations in the Darts library.

TABLE 4.1: Hyperparameter Search Spaces

Shared Search Spaces			Model-Specific Search Spaces		
Shared Hyperparameters		Search Space	TCN Hyperparameters		Search Space
Batch Size	`batch_size`	{16, 32, 64, 128}	Dilation Base	`dilation_base`	$[2, 4] \cap \mathbb{Z}$
Dropout	`dropout`	[0.1, 0.3]	Filters	`num_filters`	{32, 64, 128, 256}
Epochs	`n_epochs`	$[20, 200] \cap \mathbb{Z}$	Kernel Size	`kernel_size`	$[2, 5] \cap \mathbb{Z}$
Lag Size	`input_chunk_length`	$[1, 13] \cap \mathbb{Z}$	Weight Normalisation	`weight_norm`	{`True`, `False`}
Learning Rate	`lr`	$[1e^{-5}, 1e^{-1}]$			
			LSTM Hyperparameters		Search Space
			Neurons	`hidden_dim`	$[1, 100] \cap \mathbb{Z}$
			Layers	`n_rnn_layers`	$[1, 3] \cap \mathbb{Z}$
			Transformer Hyperparameters		Search Space
			Attention Heads	`nhead`	{1, 2, 4, 8}
			Decoder Layers	`num_decoder_layers`	$[1, 6] \cap \mathbb{Z}$
			Dimensionality	`d_model`	{32, 64, 128, 256}
			Encoder Layers	`num_encoder_layers`	$[1, 6] \cap \mathbb{Z}$
			Hidden Layer Size	`dim_feedforward`	{128, 256, 512, 1024, 2048}

4.5 Findings and Implications

This chapter concludes Phase 4: Solution Formulation of the MPSM framework by formulating the experimental setup of the study and addressing research questions 6 and 7. This provides the foundation for Phase 5: Solution Selection.

6. What forecasting models and configurations are selected for comparative evaluation for the HNL use-case?

Section 4.2 outlines the models included in the experimental setup of this study. This selection comprises two exponential smoothing models, a Croston-based method, ARIMA, and three deep neural networks. Findings from Chapter 3 form the foundation of the model selection. Moreover, Section 4.4 presents the hyperparameter tuning process adopted in the experimental setup, including the optimisation algorithm and the defined configuration space. Last, it is important to note that the data is preprocessed as described in Section 4.1.

7. How can forecasts be evaluated and validated to ensure reliable and accurate model selection for the HNL use-case?

A comprehensive description of the evaluation and validation procedures adopted in this study is provided in Section 4.3. Due to the temporal dependence inherent in time series data, careful selection of validation procedures is essential to prevent information "leakage" from future to past, leading to over-optimistic performance estimates. As detailed in this section, a nested cross validation procedure with rolling-origin-recalibration is adopted in this research, and the ME, MAE, and MASE are used as evaluation metrics.

Chapter 5

Numerical Results and Discussion

This chapter executes the experimental setup as described in Chapter 4 and presents the numerical results contributing to phase 5 of the MPSM framework: Solution Selection. As outlined in Section 1.4, research questions 8 and 9 are addressed. Section 5.1 details the performance of the forecasting models on LLI-1, LLI-2, and LLI-3, and Section 5.4 outlines the computational complexity of the validation procedure for all models.

5.1 Univariate Forecasting Performance

This section presents the numerical results following the experimental execution of the univariate models (see Figure 4.1a). The 6-month ahead forecasting performance of the models outlined in Section 3.2 is evaluated on each of the identified generic LLIs, as summarised in Table 2.3. We focus primarily on MAE in this section, since we are interested in the magnitude of forecast errors; ME reflects only bias direction, while MASE is more suited for comparisons across different time series.

5.1.1 Performance Evaluation: Long-Lead Item 1

The mean performance across all validation folds is presented in Table 5.1. We observe a negative in-sample bias (ME) for SES, DES, ARIMA, and Transformer, whereas the other models exhibit positive in-sample biases, indicating a tendency to underestimate the training data for those models. Out of sample, most models show more significant negative biases, suggesting a general tendency to overpredict unseen demand. This is mainly caused by significant demand peaks in the training data, which distort the ME, as depicted in Figure 5.1. Particularly, the deep learning models tend to react sharply to the zero demand occurrences and sporadic demand peaks. However, they often fail to match the peaks, resulting in substantial forecast biases. Conversely, the statistical models produce smoother forecasts, leading to smaller forecast biases on average — with the exception of ARIMA, which significantly overestimates demand as depicted in Figure 5.1.

Interestingly, all models perform better out-of-sample than in-sample on this dataset. This seems counter-intuitive, as one would expect a better fit to the training data than to unseen data. This occurs because the time series begins with highly volatile demand (see Figure 5.1), which consistently affects the expanding training window used in the walk-forward cross-validation across all validation folds, leading to poor in-sample performance. However, as the series stabilises over time, the models benefit from more consistent data in the (fixed-size) test set, resulting in better out-of-sample performance.

ARIMA and TCN exhibit in-sample MASE values greater than 1, indicating worse average training fit than the one-step naive forecast. The exponential smoothing methods, SBA, LSTM,

and Transformer show improved average in-sample fit to the one-step naive baseline of MASE. In terms of mean in-sample MAE, particularly ARIMA and TCN indicate poor average training fit. Moreover, we observe considerably better out-of-sample performance than the in-sample one-step naive MASE baseline for all models, as reflected by MASE values significantly less than one. This is primarily due to the training data containing more extreme fluctuations and demand peaks than the test data, leading to relatively higher in-sample errors. In terms of out-of-sample MAE, we observe the worst performance for TCN, Transformer, and ARIMA. Interestingly, SBA is the only model outperforming the RW benchmark — which ranks second — on this time series in terms of out-of-sample MAE. As expected, SES outperforms DES both in-sample and out-of-sample, which aligns with the absence of a significant linear trend in the demand pattern of LLI-1, as identified in Section 2.4.2.

TABLE 5.1: Mean Performance of the Univariate Models on the LLI-1 dataset

	In-Sample (μ)			Out-of-Sample (μ)			
Model	ME	MAE	MASE	ME	MAE	MASE	Rank
RW	0.104	15.176	1.000	0.000	8.720	0.644	2
SES	-3.751	12.557	0.861	-2.296	9.089	0.631	3
DES	-7.513	14.481	0.992	1.340	9.564	0.655	4
SBA	0.918	11.299	0.774	-0.830	**7.178**	0.522	1
ARIMA	-7.381	14.938	1.022	-6.865	10.417	0.706	6
LSTM	5.967	13.310	0.935	3.296	9.676	0.690	5
TCN	5.042	15.571	1.083	2.544	14.049	0.910	8
Transformer	-3.163	12.374	0.889	-4.224	10.768	0.739	7

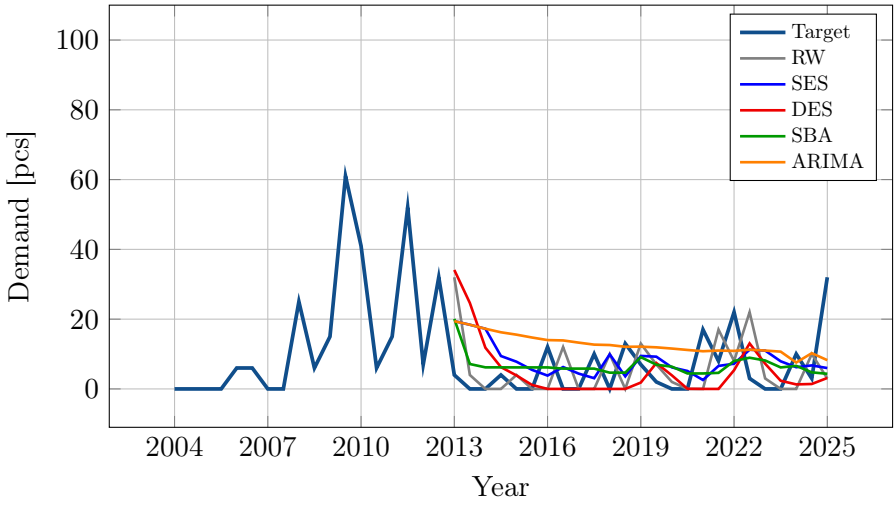
Note: Model rankings are based on mean out-of-sample MAE; the best-performing model considering mean out-of-sample MAE is shown in bold font.

Table 5.2 presents the standard deviations across all validation folds for all evaluation metrics, providing insights into the robustness and consistency of the models. In-sample, SBA exhibits the lowest variability in terms of MAE, while the RW benchmark the highest. We observe the lowest out-of-sample MAE for SBA, indicating the most stable prediction accuracy. ARIMA also exhibits relatively robust out-of-sample MAE performance, despite its comparatively poor average performance. The RW benchmark, DES, and LSTM show comparative and relatively moderate out-of-sample robustness, whereas the other ML models — particularly TCN — exhibit significant out-of-sample MAE variances. This indicates more inconsistent errors from the ML models across validation folds relative to the statistical models on this time series.

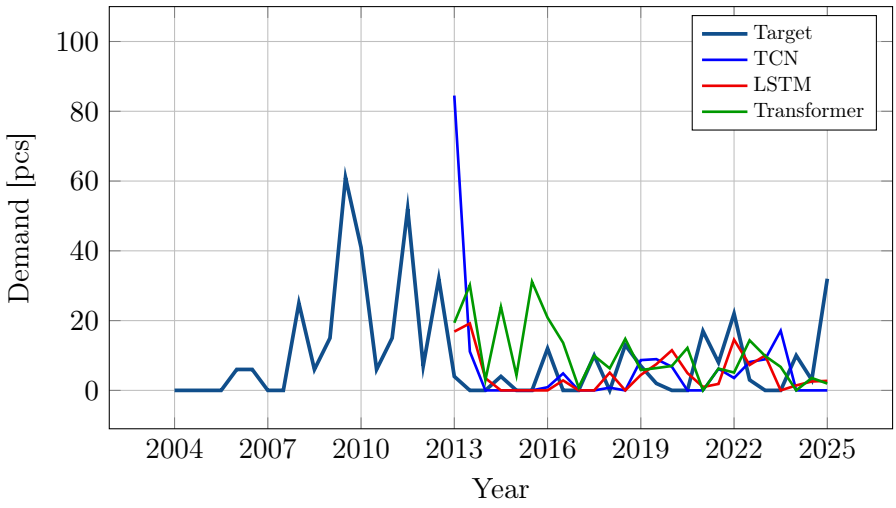
TABLE 5.2: Standard Deviation of the Performance of the Univariate Models on the LLI-1 dataset

	In-Sample (σ)			Out-of-Sample (σ)			
Model	ME	MAE	MASE	ME	MAE	MASE	Rank
RW	1.373	5.246	0.000	11.798	7.947	0.623	5
SES	3.407	2.448	0.108	10.453	5.652	0.458	3
DES	2.609	2.891	0.131	12.764	8.559	0.602	6
SBA	5.718	2.268	0.104	8.935	**5.385**	0.485	1
ARIMA	2.120	2.910	0.120	9.643	5.621	0.411	2
LSTM	10.021	4.742	0.347	11.901	7.673	0.615	4
TCN	11.261	4.996	0.311	20.767	15.504	0.770	8
Transformer	5.341	3.378	0.287	13.564	9.267	0.653	7

Note: Model rankings are based on out-of-sample MAE standard deviation; the best-performing model considering out-of-sample MAE standard deviation is shown in bold font.



(A) Forecasts by statistical models



(B) Forecasts by ML models

FIGURE 5.1: Graphical representation of demand forecasts from univariate statistical (a) and ML models (b) for LLI-1

5.1.2 Performance Evaluation: Long-Lead Item 2

Table 5.3 presents the mean performance metrics across cross-validation folds. With the exception of DES and TCN, all models exhibit negative out-of-sample bias (ME), indicating that these models tend to overestimate demand on unseen data. Figure 5.2 illustrates the actual demand and corresponding forecasts. We observe high demand peaks in the training data followed by relatively low demand sizes for LLI-2. Consequently, we observe negative biases for most models. Moreover, ARIMA fails to stabilise quickly after the sudden decrease in demand around 2012, leading to significant overestimation during that period as the model slowly adjusted to the observed demand pattern. Last, particularly the ML model Transformer exhibits significant negative bias, and thus tends to overestimate the test data. SBA and LSTM are the only models exhibiting lower bias than the RW benchmark.

Similar to the forecasts for LLI-1, all models perform better out-of-sample than in-sample. This is again due to the highly volatile demand at the start of the time series (see Figure 5.2), which consistently affects the expanding training windows. The test sets, on the other hand, lie in more stable periods, allowing for stronger out-of-sample performance.

Interestingly, the RW benchmark achieves the lowest in-sample MAE, indicating the best fit to the training data, followed closely by LSTM, then SBA, TCN, SES, DES, Transformer, and, finally, ARIMA, which fits the training data significantly worse compared to the other models. Although SBA does not achieve the best fit to the training data, it generalises best to unseen data, achieving the lowest out-of-sample MAE among all models. Along with LSTM, it is the only model to outperform the RW benchmark in terms of out-of-sample MAE. In line with the significant trend identified in Section 2.4.2, DES outperforms SES in terms of out-of-sample MAE, ranking fifth and sixth respectively. ARIMA shows the worst performance among all models, ranking eighth.

TABLE 5.3: Mean Performance of the Univariate Models on the LLI-2 dataset

	In-Sample (μ)			Out-of-Sample (μ)			
Model	ME	MAE	MASE	ME	MAE	MASE	Rank
RW	-6.387	17.271	1.000	-0.962	11.885	0.743	3
SES	-16.566	21.180	1.225	-5.944	13.013	0.740	6
DES	-16.366	22.827	1.327	3.152	12.739	0.751	5
SBA	-10.763	19.095	1.126	-4.453	**11.193**	0.662	1
ARIMA	-21.761	30.051	1.738	-8.461	16.682	0.916	8
LSTM	0.864	17.356	1.029	-0.749	11.677	0.695	2
TCN	-7.057	19.391	1.159	1.673	12.533	0.747	4
Transformer	-15.445	22.872	1.376	-5.681	14.719	0.839	7

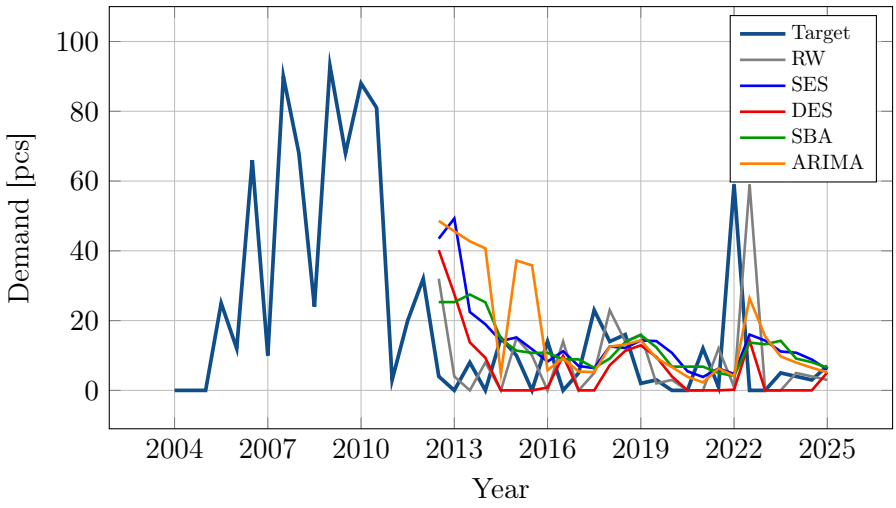
Note: Model rankings are based on mean out-of-sample MAE; the best-performing model considering mean out-of-sample MAE is shown in bold font.

The standard deviations of MAE, presented in Table 5.4, provide insight into the robustness and consistency of each model's performance across validation folds. We observe the highest in-sample standard deviations for LSTM and ARIMA, indicating significant variability in how well the models fit the training data across validation folds. Conversely, the other statistical models, TCN, and Transformer show more robust in-sample performance. Out-of-sample, SBA, LSTM, and DES show the most consistent performance, whereas the RW benchmark, ARIMA, and Transformer exhibit the least consistent errors among all models. SBA achieved the lowest out-of-sample mean MAE (see Table 5.3), indicating strong generalisation. Additionally, its standard deviation ranks first among all models, suggesting relatively superior robustness across validation folds. The Transformer model shows the least robust out-of-sample performance.

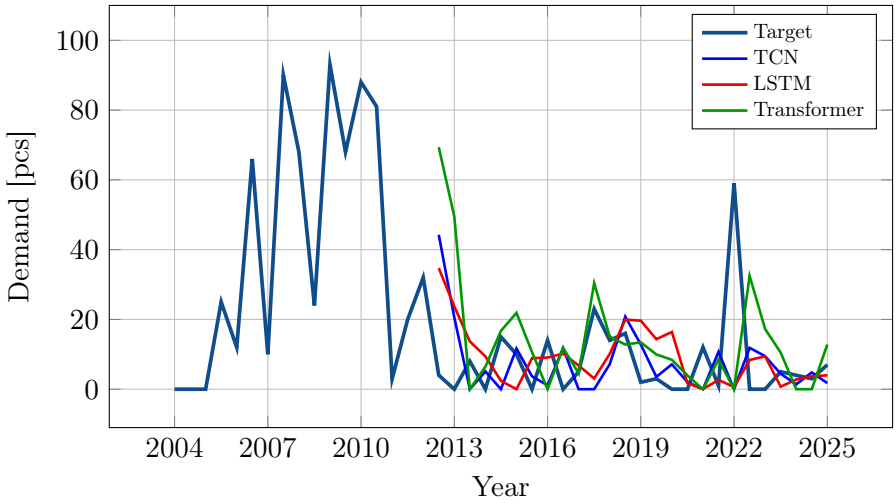
TABLE 5.4: Standard Deviation of the Performance of the Univariate Models on the LLI-2 dataset

	In-Sample (σ)			Out-of-Sample (σ)			
Model	ME	MAE	MASE	ME	MAE	MASE	Rank
RW	3.973	4.208	0.000	19.054	14.924	1.074	6
SES	5.477	5.456	0.116	17.872	13.616	0.805	4
DES	9.754	5.416	0.148	17.677	12.654	0.869	3
SBA	6.611	3.478	0.151	15.246	**11.268**	0.788	1
ARIMA	8.075	8.029	0.219	21.529	16.026	0.900	7
LSTM	14.329	7.191	0.465	16.804	12.106	0.859	2
TCN	14.304	6.221	0.391	18.819	14.138	0.964	5
Transformer	8.223	5.309	0.352	21.786	17.037	0.957	8

Note: Model rankings are based on out-of-sample MAE standard deviation; the best-performing model considering out-of-sample MAE standard deviation is shown in bold font.



(A) Forecasts by statistical models



(B) Forecasts by ML models

FIGURE 5.2: Graphical representation of demand forecasts from univariate statistical (a) and ML models (b) for LLI-2

5.1.3 Performance Evaluation: Long-Lead Item 3

Table 5.5 provides a summary of the average forecast error metrics on the LLI-3 time series across all cross-validation folds. We find that all models — excluding the RW benchmark — exhibit a positive out-of-sample bias (ME), indicating a general tendency to underestimate the target series. Particularly SBA shows a relatively high positive bias both in- and out-of-sample compared to the other models, whereas the exponential smoothing models produce relatively unbiased forecasts on the out-of-sample data.

The statistical models — excluding the naive RW benchmark — perform relatively well on the LLI-3 dataset, with four of the lowest out-of-sample MAE values observed. Interestingly, despite the significant trend identified in Section 2.4.2, the SES model outperforms DES. This can be attributed to the lumpy demand pattern of the time series. While DES tries to capture the underlying trend, sporadic demand shifts can lead to unstable trend estimates, resulting in more volatile forecasts than SES. In contrast, SES offers more stable predictions by smoothing out these irregular changes as it only includes a level component, as illustrated in Figure 5.3. While LSTM performs best in-sample in terms of MAE — suggesting that the model fits the training

data relatively well — forecasts on the unseen data have a relatively high MAE, indicating poor generalisation. Moreover, we observe poor performance from TCN, which fails to effectively fit the training data, resulting in the worst out-of-sample forecasts among all models. Notably, it is the only model that is outperformed by the RW benchmark on this dataset. SES outperformed all models out-of-sample, despite not achieving the best in-sample performance.

TABLE 5.5: Mean Performance of the Univariate Models on the LLI-3 dataset

	In-Sample (μ)			Out-of-Sample (μ)			
Model	ME	MAE	MASE	ME	MAE	MASE	Rank
RW	1.264	16.762	1.000	-0.080	27.120	1.641	7
SES	2.001	13.028	0.793	7.723	**19.545**	1.194	1
DES	-1.634	13.619	0.828	4.830	20.086	1.227	2
SBA	5.145	12.369	0.750	9.566	20.235	1.221	3
ARIMA	2.641	13.230	0.811	6.540	20.364	1.261	4
LSTM	3.876	12.267	0.725	6.416	22.138	1.379	6
TCN	1.333	13.301	0.742	9.198	28.259	1.604	8
Transformer	-2.649	13.679	0.792	5.816	21.154	1.246	5

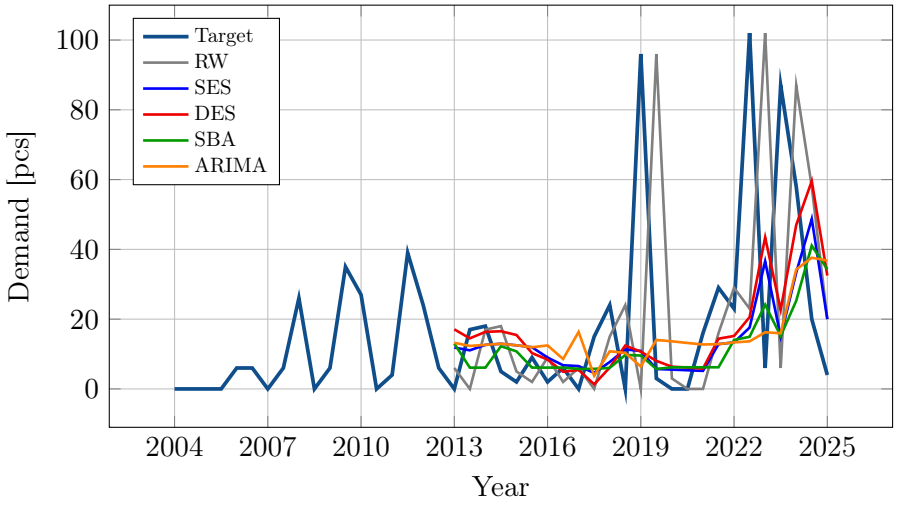
Note: Model rankings are based on mean out-of-sample MAE; the best-performing model considering mean out-of-sample MAE is shown in bold font.

Table 5.6 reports the standard deviation of forecast error metrics across all cross-validation folds for each model, providing insights into the consistency and reliability of model performance. In general, the statistical models — SES, DES, SBA, and ARIMA — exhibit relatively low variability in both in-sample and out-of-sample MAE, indicating more consistent and stable forecasting performance across cross-validation folds when compared to the ML models. ARIMA demonstrates the lowest in-sample standard deviation, while DES the lowest out-of-sample standard deviation, indicating superior robustness. In contrast, we observe substantial out-of-sample variability for RW and TCN, confirming their unstable fit to the training data. In addition to having the worst mean performance, these models also show the highest variance in forecast errors across cross-validation folds, indicating it are the least robust and reliable models overall for this time series. Figure 5.3 confirms with visually more smoothed forecasts that statistical models prioritise level and trend over short-term fluctuations, in contrast to the more volatile outputs of ML models.

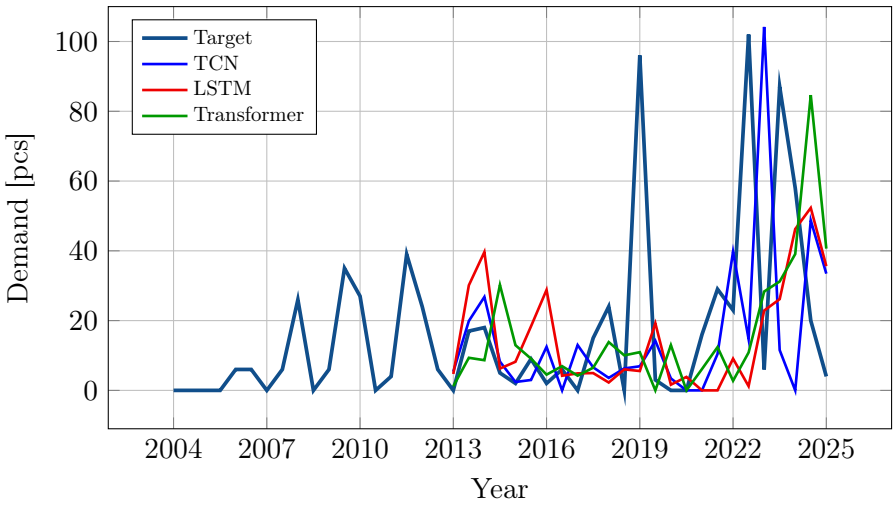
TABLE 5.6: Standard Deviation of the Performance of the Univariate Models on the LLI-3 dataset

	In-Sample (σ)			Out-of-Sample (σ)			
Model	ME	MAE	MASE	ME	MAE	MASE	Rank
RW	1.440	4.952	0.000	42.157	32.275	2.041	7
SES	2.845	3.098	0.068	29.803	23.788	1.600	2
DES	2.845	3.216	0.071	30.462	**23.405**	1.589	1
SBA	3.066	3.222	0.069	30.057	24.197	1.626	4
ARIMA	3.100	3.032	0.077	30.926	24.177	1.655	3
LSTM	6.035	5.589	0.201	33.005	25.306	1.740	6
TCN	5.749	9.779	0.396	43.050	33.754	1.879	8
Transformer	8.838	7.914	0.255	32.466	25.305	1.643	5

Note: Model rankings are based on out-of-sample MAE standard deviation; the best-performing model considering out-of-sample MAE standard deviation is shown in bold font.



(A) Forecasts by statistical models



(B) Forecasts by ML models

FIGURE 5.3: Graphical representation of demand forecasts from univariate statistical (a) and ML models (b) for LLI-3

5.2 Multivariate Forecasting Performance

This section presents the numerical results following the experimental execution of the multivariate models (see Figure 4.1b). These multivariate models incorporate project-related covariate time series and temporal encodings, as detailed in Section 4.1.2. The 6-month ahead forecasting performance of the multivariate ML models, as outlined in Section 3.2, is evaluated on each of the identified generic LLIs. Again, we focus primarily on MAE in this section, since we are interested in the magnitude of forecast errors.

5.2.1 Performance Evaluation: Long-Lead Item 1

Table 5.7 summarises the average forecast error metrics on the LLI-1 time series across all cross-validation folds. While the LSTM model shows the best in-sample fit among all (univariate and multivariate) models, no improved mean out-of-sample performance is observed for the multivariate models compared to the univariate models. Notably, TCN is the only model showing an out-of-sample improvement, despite exhibiting a poor in-sample fit. The model significantly overestimates training demand, producing inflated fitted values and resulting in large in-sample errors. This suggests that the model struggles to use the past covariates effectively. Transformer, on the other hand, shows poor performance across both settings. Overall, none of the multivariate models prove competitive with the best-performing univariate models on this time series.

TABLE 5.7: Mean Performance of the Multivariate Models on the LLI-1 dataset

	In-Sample (μ)			Out-of-Sample (μ)			
Model	ME	MAE	MASE	ME	MAE	MASE	Rank
LSTM	6.774	10.696 (-19.6%)	0.757	-0.328	**10.862** (+41.6%)	0.758	1
TCN	-47.889	60.578 (+289%)	4.110	-3.507	11.448 (-18.5%)	0.765	2
Transformer	-3.263	16.548 (+33.7%)	1.188	-3.070	13.311 (+23.6%)	0.911	3

Note: The percentages in parentheses indicate the relative MAE difference compared to the univariate version of each model.

Table 5.8 confirms the inflated in-sample fitted values for TCN, reflected in its large standard deviation. While the multivariate LSTM shows competitive robustness in-sample, its out-of-sample MAE standard deviation indicates poor generalisation relative to its univariate variant. Transformer shows no benefit from the covariate data in this dataset, based on both in-sample and out-of-sample MAE mean and standard deviation. Figure 5.4 visually confirms the unstable out-of-sample predictions of Transformer, with pronounced peaks.

TABLE 5.8: Standard Deviation of the Performance of the Multivariate Models on the LLI-1 dataset

	In-Sample (σ)			Out-of-Sample (σ)			
Model	ME	MAE	MASE	ME	MAE	MASE	Rank
LSTM	4.619	2.494 (-47.4%)	0.214	14.377	9.424 (+22.8%)	0.649	2
TCN	42.003	43.077 (+762%)	2.527	13.852	**8.551** (-44.8%)	0.637	1
Transformer	6.381	6.724 (+99.1%)	0.591	19.421	14.471 (+56.2%)	0.919	3

Note: The percentages in parentheses indicate the relative MAE standard deviation difference compared to the univariate version of each model.

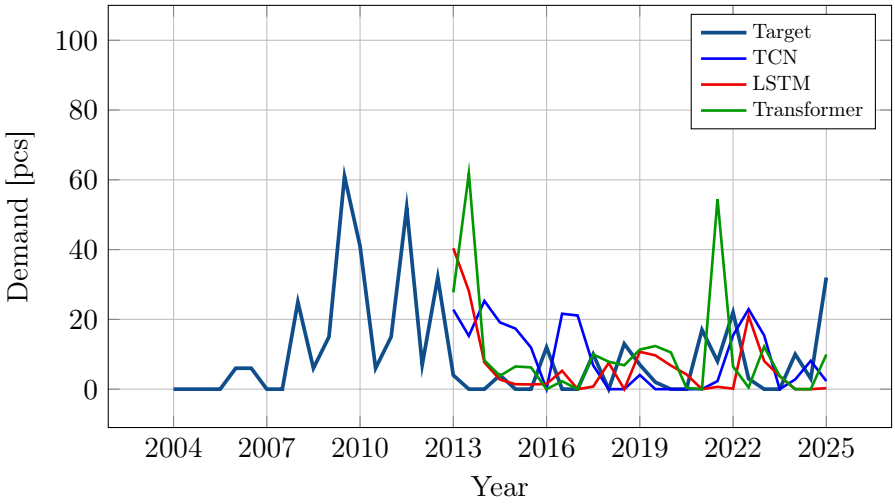


FIGURE 5.4: Graphical representation of demand forecasts from multivariate ML models for LLI-1

5.2.2 Performance Evaluation: Long-Lead Item 2

The average forecast error metrics on the LLI-2 time series is depicted in Table 5.9. We observe poor in-sample fit for TCN and Transformer relative to their univariate variants, suggesting that the covariates do not improve fit. However, Transformer generalises better out-of-sample compared to the univariate Transformer model, while TCN does not show any improvement. LSTM, on the other hand, already showed competitive performance in the univariate case and now benefits further from the covariates, achieving the best performance among all univariate and multivariate models on the LLI-2 time series. In addition to improved out-of-sample MAE, it also produced the most unbiased estimates of unseen demand.

TABLE 5.9: Mean Performance of the Multivariate Models on the LLI-2 dataset

	In-Sample (μ)			Out-of-Sample (μ)			
Model	ME	MAE	MASE	ME	MAE	MASE	Rank
LSTM	4.074	15.541 (-10.5%)	0.926	0.108	**10.981** (-6.0%)	0.717	1
TCN	-67.331	74.818 (+286%)	4.666	-6.964	17.746 (+41.6%)	1.049	3
Transformer	-11.842	23.497 (+2.7%)	1.463	-4.038	13.889 (-4.7%)	0.837	2

Note: The percentages in parentheses indicate the relative MAE difference compared to the univariate version of each model.

The standard deviations in Table 5.10 indicate unstable predictions for the multivariate TCN model, confirming its poor performance on this time series. Transformer shows reduced robustness in-sample but is more consistent out-of-sample than its univariate variant, though still not competitively stable. Last, multivariate LSTM not only improves mean MAE, but also shows a significant increase in in-sample robustness. Out-of-sample, it is slightly less robust but remains moderately competitive.

TABLE 5.10: Standard Deviation of the Performance of the Multivariate Models on the LLI-2 dataset

	In-Sample (σ)			Out-of-Sample (σ)			
Model	ME	MAE	MASE	ME	MAE	MASE	Rank
LSTM	10.274	4.316 (-40.0%)	0.269	17.178	**13.211** (+9.1%)	0.989	1
TCN	56.025	54.066 (+769%)	3.428	24.244	17.926 (+26.8%)	1.160	3
Transformer	9.494	7.863 (+48.1%)	0.589	20.287	15.329 (-10.0%)	1.109	2

Note: The percentages in parentheses indicate the relative MAE standard deviation difference compared to the univariate version of each model.

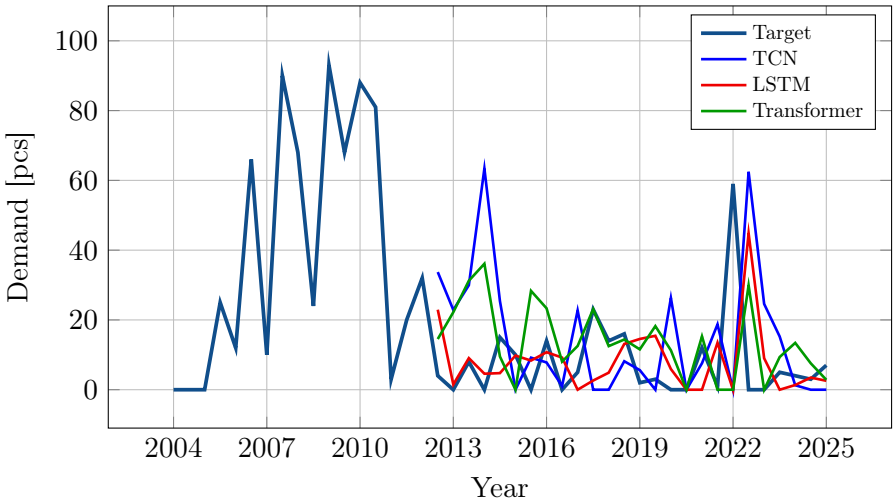


FIGURE 5.5: Graphical representation of demand forecasts from multivariate ML models for LLI-2

5.2.3 Performance Evaluation: Long-Lead Item 3

The inclusion of covariate data leads to reduced in-sample mean MAE values for LSTM and Transformer, but it does not lead to improved performance on unseen data in terms of mean

MAE. The opposite is true for TCN on this time series, which shows poor in-sample fit but benefits from covariates out-of-sample, achieving better generalisation than its univariate variant. However, the multivariate TCN model still fails to achieve competitive out-of-sample performance on the LLI-3 time series, as it is outperformed by all univariate models except the RW benchmark and its own univariate variant. We conclude that none of the multivariate models are competitive with the best-performing univariate models on the LLI-3 series.

TABLE 5.11: Mean Performance of the Multivariate Models on the LLI-3 dataset

	In-Sample (μ)			Out-of-Sample (μ)			
Model	ME	MAE	MASE	ME	MAE	MASE	Rank
LSTM	6.508	12.144 (-1.0%)	0.739	6.077	25.485 (+15.1%)	1.481	2
TCN	-20.756	36.386 (+174%)	2.360	3.443	**23.480** (-16.9%)	1.487	1
Transformer	-0.494	14.556 (+6.4%)	0.869	8.503	28.245 (+33.5%)	1.632	3

Note: The percentages in parentheses indicate the relative mean MAE difference compared to the univariate version of each model.

Table 5.12 reports the standard deviation of forecast error metrics across all cross-validation folds for each model. Despite more consistent in-sample MAE values for LSTM and Transformer, and more consistent out-of-sample MAE values for TCN, none of the models demonstrate improved robustness over the most robust univariate models. This again indicates that the formulated covariates offer no advantages on this time series.

TABLE 5.12: Standard Deviation of the Performance of the Multivariate Models on the LLI-3 dataset

	In-Sample (σ)			Out-of-Sample (σ)			
Model	ME	MAE	MASE	ME	MAE	MASE	Rank
LSTM	4.288	4.480 (-19.8%)	0.208	45.306	37.948 (+50.0%)	2.196	3
TCN	24.807	25.185 (+158%)	2.090	33.840	**24.611** (-27.1%)	1.681	1
Transformer	5.381	6.333 (-20.0%)	0.246	41.319	31.333 (+23.8%)	1.862	2

Note: The percentages in parentheses indicate the relative MAE standard deviation difference compared to the univariate version of each model.

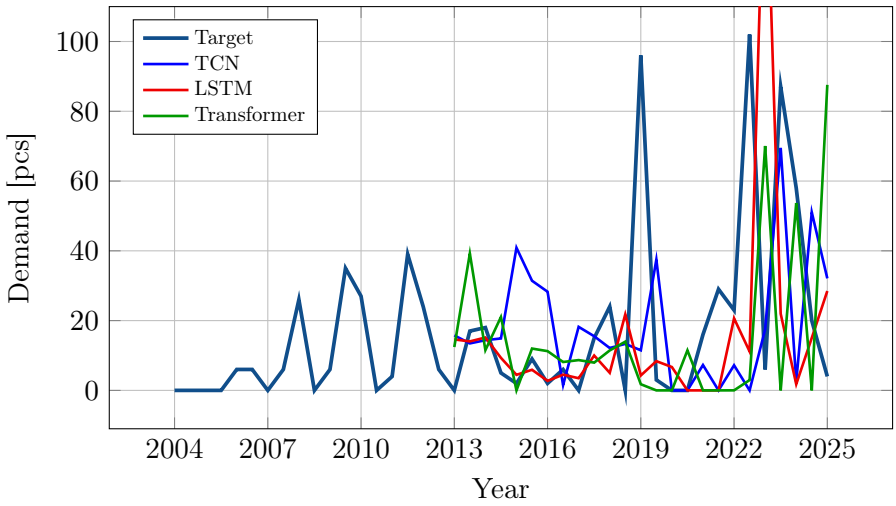


FIGURE 5.6: Graphical representation of demand forecasts from multivariate ML models for LLI-3

5.3 Forecastibility Comparison

This section assesses the relative forecastability of the three generic LLIs by focusing on out-of-sample MASE values. As a scale-independent metric, MASE allows for comparison across time series, regardless of the absolute magnitude of demand (see Section 3.2). In addition to the MASE values, we also consider model performance relative to the naive RW benchmark. The extent to which models are able to outperform this baseline provides insight into how easy or difficult it is to capture patterns in each time series — and thus into the relative forecastability of each LLI.

Figure 5.7 presents the out-of-sample MASE values for the best-performing model in terms of out-of-sample MAE and the RW benchmark for each LLI. The results reveal notable differences in forecastability:

- LLI-1 is the most forecastable item, with the best model achieving a mean MASE of 0.522 — notably lower than the benchmark value of 0.644. This considerable performance improvement (-18.9%) suggests that there are learnable patterns in the time series that the model is able to exploit. Moreover, the relatively low standard deviation in MASE across validation folds indicates that the model's performance is consistent and robust for this item compared to the other LLIs. As illustrated in Figure 2.4, this LLI exhibits the lowest CV of positive demand sizes, supporting its forecastibility despite a higher ADI compared to LLI-2 and LLI-3.

- LLI-2 shows relatively moderate forecastability, with the best model reaching a MASE of 0.717, compared to 0.743 for the benchmark. While this performance improvement (-3.5%) is modest — especially relative to LLI-1 — it still suggests the presence of some predictable structure in the time series. However, the improvement over the naive RW benchmark is minimal, and the relatively high standard deviation across validation folds indicates poor robustness.

- LLI-3 appears significantly less forecastable. The best model obtains a MASE of 1.194, while the RW benchmark reaches a much higher value of 1.641. Interestingly, this constitutes the largest performance gain over the naive RW benchmark (-37.4%) among the three LLIs. However, the overall magnitude of the error remains high — with both MASE values well above 1 — suggesting that forecasts are still worse than using a one-step naive

approach on the training data. This, combined with the relatively large variation across validation folds, indicates that LLI-3's time series contains fewer consistent or exploitable patterns, making it inherently more difficult to forecast accurately. This is caused by pronounced peak demands, as illustrated in Figure 2.3c.

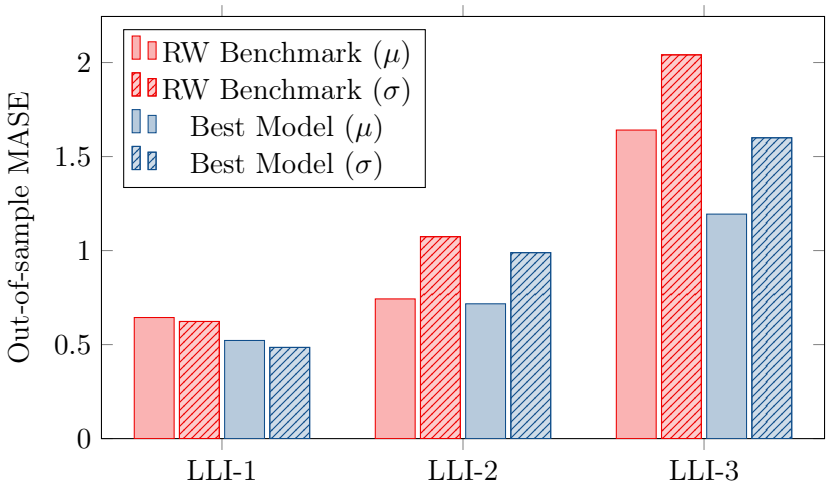


FIGURE 5.7: Comparison of mean (μ) and standard deviation (σ) of out-of-sample MASE values between the best-performing model and the naive RW benchmark for each LLI.

5.4 Computational Complexity

As mentioned in Section 4.4.1, a drawback of our validation procedure is its computational complexity. This section reports the computational times for our experiments to provide insights into the complexity of the validation procedure and the relative computational times of the forecasting models. Note that these computational times relate to the entire validation process, which estimates the expected generalisation performance of a model. This is not the computational time required to obtain a forecast in practice, as the validation procedure effectively simulates how each model would have performed if it had been used historically. Experiments were run on a server with an AMD EPYC 7713 64-core CPU, with 3.675 GHz and 1 TB of RAM.

Table 5.13 presents the average computational time (in hours) per model across all LLIs. This includes the full experiments, including nested-cross validation with rolling-origin-recalibration and hyperparameter tuning, where 200 trials were used as the stopping criterion. Unsurprisingly, the simple naive benchmark RW is the least computationally expensive, as it has no parameters to optimise. Among the remaining models, we observe the lowest average computational time for ARIMA, which employs AICc for automatic modelling using `AutoARIMA`, and therefore does not require TPE hyperparameter tuning. DES requires slightly more computational time than SES and SBA due to the inclusion of a trend component, requiring optimisation. Moreover, the deep learning models — LSTM, TCN, and Transformer — incur substantially higher computational costs due to their complex architectures. Particularly Transformer requires significant computational time, likely because this model benefits greatly from parallel processing due to its self-attention mechanism, which is not adopted in this study. Interestingly, the multivariate versions of LSTM and TCN require less computational time. This indicates that for these multivariate models, less complex neural network architectures are found to perform best on the validation set relative to the univariate models. The multivariate Transformer requires slightly longer computational time during nested cross-validation than its univariate variant.

TABLE 5.13: Average computational time (in hours) per model for the nested cross-validation procedure, including hyperparameter tuning, averaged across all LLIs

Univariate Models				Multivariate Models	
Model	Time (hrs)	Model	Time (hrs)	Model	Time (hrs)
RW	0.001	ARIMA	0.019	LSTM	5.713
SES	0.481	LSTM	10.222	TCN	5.379
DES	0.546	TCN	7.854	Transformer	19.617
SBA	0.493	Transformer	19.546		

5.5 Findings and Implications

This chapter concludes Phase 5: Solution Selection of the MPSM framework by performing the experimental execution and discussing the numerical results, addressing research question 8 and research question 9.

8. Which forecasting model performs best for the selected LLIs in the HNL use-case?

Section 5.1 detailed the cross-validated performances of all models on each of the generic LLIs. Key findings include:

- Outperforming the demand forecast from the naive RW benchmark proved surprisingly difficult for LLI-1, with only SBA outperforming the RW benchmark in terms of mean out-of-sample MAE. Notably, the complex LSTM, ARIMA, Transformer and TCN exhibited poor average performance. Moreover, the RW benchmark, DES, and the ML models — particularly TCN — showed lower robustness across validation folds compared to the other statistical models. In addition to achieving the lowest average out-of-sample MAE, SBA also exhibits the highest robustness across validation folds and remains relatively unbiased. Last, none of the multivariate models prove competitive with the best-performing univariate models on this time series.

- For LLI-2, the multivariate LSTM model exhibited the best in-sample fit and generalised the best to unseen data, achieving the best out-of-sample MAE. The multivariate LSTM was, along with its univariate variant and SBA, the only model to outperform the RW benchmark in terms of out-of-sample MAE. The multivariate TCN exhibited the highest out-of-sample MAE of all models, followed by ARIMA. In terms of robustness, SBA and univariate LSTM demonstrated the most consistent out-of-sample errors, whereas the multivariate variant of TCN showed the least robustness across validation folds. The multivariate LSTM model — which achieved the best mean out-of-sample performance — exhibited relatively moderate robustness.

- SES achieved the best forecasting performance for LLI-3, outperforming all other models in both in-sample and out-of-sample MAE. SES was followed by DES and SBA in terms of out-of-sample MAE. This suggests that despite the presence of a trend, SES outperformed DES, likely due to its more stable estimates in the erratic demand pattern. ARIMA also delivered competitive performance, while all univariate and multivariate ML models showed poor generalisation in terms of out-of-sample MAE. All models except the univariate TCN and multivariate Transformer outperformed the RW benchmark in terms of out-of-sample MAE, indicating that it was relatively easy to outperform on this LLI. Moreover, DES and SES showed the most robustness in forecast errors, while the RW benchmark and ML models, particularly multivariate LSTM and univariate TCN, exhibited significantly higher variance, indicating less consistent performance across validation folds.

9. How do the selected models compare in terms of computational complexity?

As expected, the naive RW benchmark was the least computationally expensive, as it requires no parameter optimisation. Among the remaining models, ARIMA had the lowest average runtime, since `AutoARIMA` selects parameters automatically using AICc, without requiring TPE-based hyperparameter tuning. SES and SBA were also relatively efficient, while DES took slightly longer due to the additional trend component. In contrast, the ML models — LSTM, TCN, and Transformer — incurred substantially higher computational costs due to their complex architectures and expensive tuning procedures. Transformer was the most computationally expensive model overall, likely because this model benefits greatly from parallel processing due to its self-attention mechanism, which is not adopted in this study.

Chapter 6

Conclusions and Recommendations

This section concludes the research by addressing the final phase of the MPSM framework — Evaluation — related to research question 10. It presents the key findings, implications, and recommendation for the problem owner and further research, and concludes by answering the main research question:

To what extent is it feasible to forecast generic LLI demand in HNL's ETO production environment, and to what extent can this enable forecast-driven procurement?

Section 6.1 summarises the key findings of this study, and Section 6.2 outlines practical, actionable recommendations for the problem owner. The main academic and practical contributions are presented in Section 6.3. Last, Section 6.4 discusses the limitations of the research and proposes future research directions.

6.1 Conclusions

A team of buyers reactively procures LLIs once basic engineering is finished, resulting in frequent late deliveries of LLIs. Approximately 34% of all purchase orders were delivered on or before their planned delivery date over the past four years, 74 days too late on average. This research aimed to evaluate demand forecasting models to assess the feasibility of accurately predicting demand for generic LLIs in HNL's ETO production environment, potentially enabling earlier procurement. We identified three potentially suitable generic bearing LLIs for earlier procurement, referred to as LLI-1, LLI-2, and LLI-3 for brevity. To support timely procurement decision-making, a forecast horizon equal to six months — comprising the five-month average lead time and an additional one-month review period — was used for each generic LLI, with time buckets of six months.

A literature study provided insight into existing demand forecast models. Although demand forecasting has been studied extensively over the past decades, mixed and limited evidence is available about the relative performance of statistical and ML models in terms of accuracy and computational requirements. However, numerous studies have shown that deep learning models are generally more accurate than shallow ML models. Therefore, this study included a broad range of both statistical and deep learning models: SES, DES, SBA, ARIMA, LSTM, TCN, and Transformer, which were compared against a simple naive RW benchmark. In addition, multivariate variants of the ML models were implemented using covariate data, including temporal encodings and project-related features. Specifically, the number of signed projects and the estimated number of sheaves per class per period were used as past covariates.

A nested cross-validation approach was adopted in this study to obtain unbiased estimates of forecasting accuracy. Key findings of this study include:

- The best-performing model varied by time series. For LLI-1, the statistical SBA model clearly outperformed all others — including complex deep learning models like LSTM and Transformer — by achieving the lowest out-of-sample MAE, highest robustness, and minimal bias. SBA achieved a mean ME of -0.830, a mean MAE of 7.178, and a mean MASE of 0.522 out-of-sample.
- LLI-2 was best forecasted by the multivariate LSTM, which showed the lowest bias and best out-of-sample performance, although it was less robust than SBA and its univariate variant. The multivariate LSTM model achieved a mean ME of 0.108, a mean MAE of 10.981, and a mean MASE of 0.717 out-of-sample on this time series.
- For LLI-3, SES delivered the most accurate and robust forecasts despite the item's erratic demand, outperforming all other models in terms of out-of-sample MAE. On average, it achieved a ME of 7.723, indicating relatively unbiased forecasts on unseen data, a MAE of 19.545, and a MASE of 1.194 out-of-sample.
- LLI-1 is the most forecastable generic LLI, with the best-performing model (in terms of out-of-sample MAE) showing a notable improvement over the naive RW benchmark in terms of out-of-sample MASE (-18.9%), along with the lowest mean and standard deviation of MASE.
- In contrast, LLI-2 shows only a modest improvement over the benchmark in terms of out-of-sample MASE (-3.5%) and suffers from poor robustness due to higher variability across validation folds.
- LLI-3 demonstrates the largest performance gain over the RW benchmark in terms of out-of-sample MASE (-37.4%), but both its high MASE values and substantial variability indicate limited forecastability, driven by pronounced peak demands and fewer exploitable patterns in the time series.

These results indicate that the feasibility of forecasting generic LLI demand in HNL's ETO production environment is mixed and highly dependent on the characteristics of the specific LLI. However, it proved surprisingly difficult to consistently outperform the simple naive RW benchmark. Since the objective is to evaluate the extent to which accurate demand predictions can be realised in HNL's ETO production environment, enabling earlier procurement, we conclude that forecast-based procurement is only conditionally viable. While particularly LLI-1 show promising predictive potential, others exhibit high variability or minimal gains over simple naive benchmarks. Therefore, any implementation of forecast-driven procurement should begin with the identification of items that consistently demonstrate both high forecast accuracy and practical value for early procurement — that is, items for which earlier availability has the potential to reduce overall project lead times. This should be supported by improved data collection and further model refinement.

6.2 Recommendations

Given the insights obtained in this study, we recommend the following practical actions and considerations for HNL:

- Evaluate project timelines to determine whether early procurement based on forecasts offers practical benefits. If the replenishment lead time is shorter than the time between the initiation of procurement and actual usage in the manufacturing process, the item is not on the project's critical path, and forecast-based procurement may not be necessary. It is recommended to collect more granular data on the exact timing of part usage within projects, as this information was not available in the current study but is essential to

determine whether a part lies on the project's critical path. Without such data, it is not possible to reliably assess the practical value of early procurement or its potential to reduce overall project lead times.

If specific items are identified as potentially benefiting from forecast-driven procurement based on the above evaluation, we recommend the following actions to support its successful implementation:

- Integrate the experimental forecasting setup developed in this study into an internal test environment to further explore the feasibility of forecast-based procurement, and adopt a continued focus on improving demand predictability. Refer to Section 6.4.2 for detailed future research directions.
- Prioritise consistent and detailed data collection — including project-related data (e.g., project timelines, project attributes) and part-specific demand data — as ML models depend heavily on adequate historical data availability to deliver accurate forecasts and support future procurement decisions. While ML models did not consistently outperform statistical models in this study, their performance is known to improve with larger and richer datasets, suggesting potential future benefits as data availability increases.
- If forecast-driven procurement is implemented for selected LLIs, inventory policies should be adapted accordingly. To effectively manage inventory in forecast-driven procurement, safety stock levels should be calculated based on the observed forecast accuracy. Specifically, using the MAE of out-of-sample forecasts, safety stock can be set using the MAE and an appropriate Z-score corresponding to the desired service level if we assume normally distributed forecast errors. This approach ensures safety stock reflects the variability in forecast errors, providing a buffer to mitigate stockouts in the erratic ETO production environment.

6.3 Main Contributions

This section presents the key contributions of the study from both an academic and a practical perspective. Contributions to scientific literature are outlined in Section 6.3.1, and practical contributions to HNL are given in Section 6.3.2.

6.3.1 Academic Contributions

This study advances state-of-the-art demand forecasting models for three generic LLIs. Although demand forecasting has received considerable attention in the literature, few studies addressed erratic and lumpy demand forecasting in an ETO setting. Key academic contributions include:

- A framework for identifying the most suitable items for early procurement in an ETO setting by ranking purchase groups with AHP-express and selecting generic LLIs based on a set of constraints.
- Objective and unbiased empirical evidence providing insights into the performance of statistical and deep learning one-step-ahead forecasting models in highly irregular demand environments, including a fair benchmark.

6.3.2 Practical Contributions

This study contributes to practice by identifying the most promising and impactful items for forecast-based procurement and by presenting an extensive evaluation of demand forecasting models in HNL's ETO environment, providing insights into the forecastibility of these items. The research offers promising directions to potentially realise practical applicability, and provides a structured experimental setup to support potential future forecasting efforts.

6.4 Limitations and Future Research

This section presents the limitations and future research directions of the study. Research limitations are outlined in Section 6.4.1, and future research directions to HNL are given in Section 6.4.2.

6.4.1 Research Limitations

Historical part issue data is used as and regarded as demand data. However, this may not fully reflect actual demand patterns, as part issues in the historical data do not necessarily align with the moments when the demand originally occurred. Last, this research is limited to state-of-the-art individual univariate and multivariate models, hybrid and ensemble models are not included.

6.4.2 Future Research Directions

This section details key future research directions that could address the limitations of this study and contribute to further improvements in forecast accuracy. These directions aim to further improve the forecasts to achieve real-world applicability.

- The current study researched univariate and multivariate forecasting models. Future research could explore the potential benefit of incorporating different combinations of covariate data into the multivariate models — such as project-related data, macroeconomic indicators such as oil prices, or gross wind energy capacity installed, or supplier-related data — into the forecasting models. This is particularly promising for deep learning models, which typically benefit from large amounts of input data to learn complex patterns and improve prediction accuracy.
- Future research could explore additional model variations and hybrid approaches to enhance the forecast accuracy. Moreover, combining predictions through ensemble models might complement the strengths and weaknesses of individual models, potentially improving forecast accuracy and robustness.
- CL could be applied to the deep learning models to investigate the potential of CL models. Adopting effective strategies for extracting information from large datasets into the forecasting models may lead to improved performance over traditional forecasting models.
- To reduce the computational complexity of the validation procedure, substantial improvements can be expected through GPU acceleration and parallel processing. Future research could investigate these approaches to enable more efficient model validation, facilitating the exploration of larger configuration spaces, more complex models, and increased hyperparameter optimisation.

References

Abbasimehr, H., Shabani, M., & Yousefi, M. (2020). An optimized model using LSTM network for demand forecasting. *Computers & industrial engineering*, *143*, 106435. doi: https://doi.org/10.1016/j.cie.2020.106435

Afaq, S., & Rao, S. (2020). Significance of epochs on training a neural network. *Int. J. Sci. Technol. Res*, *9*(06), 485–488.

Ali, Ö. G., Sayın, S., Van Woensel, T., & Fransoo, J. (2009). SKU demand forecasting in the presence of promotions. *Expert Systems with Applications*, *36*(10), 12340–12348.

Amin-Naseri, M. R., & Tabar, B. R. (2008). Neural network approach to lumpy demand forecasting for spare parts in process industries. In *2008 international conference on computer and communication engineering* (pp. 1378–1382).

Archer, B. H. (1980). Forecasting demand: Quantitative and intuitive techniques. *International Journal of Tourism Management*, *1*(1), 5-12. doi: https://doi.org/10.1016/0143-2516(80)90016-X

Avinash, S. S., Mohoan, V., & Ranjana, P. (2024). Real time taxi demand prediction using recurrent neural network. In *Aip conference proceedings* (Vol. 3044).

Axsäter, S. (2015). Inventory control. In (Vol. 225, p. 7-37). Springer.

Babai, M. Z., Tsadiras, A., & Papadopoulos, C. (2020). On the empirical performance of some new neural network methods for forecasting intermittent demand. *IMA Journal of Management Mathematics*, *31*(3), 281–305.

Bai, S., Kolter, J. Z., & Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, *10*. doi: https://doi.org/10.48550/arXiv.1803.01271

Barbosa, N. d. P., Christo, E. d. S., & Costa, K. (2015). Demand forecasting for production planning in a food company. *Arpn journal of engineering and applied sciences*, *10*(16), 7137–7141.

Barrow, D., Kourentzes, N., Sandberg, R., & Niklewski, J. (2020). Automatic robust estimation for exponential smoothing: Perspectives from statistics and machine learning. *Expert Systems with Applications*, *160*, 113637. doi: https://doi.org/10.1016/j.eswa.2020.113637

Basodi, S., Ji, C., Zhang, H., & Pan, Y. (2020). Gradient amplification: An efficient way to train deep neural networks. *Big Data Mining and Analytics*, *3*(3), 196–207.

Beck, N., Dovern, J., & Vogl, S. (2025). Mind the naive forecast! a rigorous evaluation of forecasting models for time series with low predictability. *Applied Intelligence*, *55*(6), 395.

Bengio, Y., Goodfellow, I., & Courville, A. (2017). *Deep learning* (Vol. 1). MIT press Cambridge, MA, USA.

Benidis, K., Rangapuram, S. S., Flunkert, V., Wang, Y., Maddix, D., Turkmen, C., ... others (2022). Deep learning for time series forecasting: Tutorial and literature survey. *ACM Computing Surveys*, *55*(6), 1–36. doi: https://doi.org/10.48550/arXiv.2004.10240

Bergmeir, C., & Benítez, J. M. (2012). On the use of cross-validation for time series predictor evaluation. *Information Sciences*, *191*, 192-213. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0020025511006773` (Data Mining for Software Trustworthiness) doi: https://doi.org/10.1016/j.ins.2011.12.028

Bernard, E. (2021). Tntroduction to machine learning. In (chap. 2). Wolfram Media, Inc.

Borovykh, A., Bohte, S., & Oosterlee, C. W. (2017). Conditional time series forecasting with convolutional neural networks. *arXiv preprint arXiv:1703.04691*.

Box, G., & Jenkings, G. (1970). Time Series Analysis: Forecasting and Control. *Holden Day, San Francisco, USA*.

Boylan, J., & Syntetos, A. (2007). The accuracy of a Modified Croston procedure. *International Journal of Production Economics*, *107*(2), 511-517. (Operations Management in China) doi: https://doi.org/10.1016/j.ijpe.2006.10.005

Brown, R. G., & Meyer, R. F. (1961). The fundamental theorem of exponential smoothing. *Operations Research*, *9*(5), 673–685. Retrieved from `https://www.jstor.org/stable/166814`

Carmo, J. L., & Rodrigues, A. J. (2004). Adaptive forecasting of irregular demand processes. *Engineering Applications of Artificial Intelligence*, *17*(2), 137–143.

Caroleo, B., Chiusano, S., Daraio, E., Avignone, A., Gastaldi, E., Paoletti, M., & Arnone, M. (2024). Machine learning methods to forecast public transport demand based on smart card validations. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, *540 LNICST*, 194 – 209. doi: https://doi.org/10.1007/978-3-031-49379-9_11

Cerqueira, V., Torgo, L., & Soares, C. (2019). Machine learning vs statistical methods for time series forecasting: Size matters. *arXiv preprint arXiv:1909.13316*.

Chopra, S., & Meindl, P. (2007). Supply chain management. strategy, planning & operation. In (p. 189-220). Springer.

Cichy, R. M., & Kaiser, D. (2019). Deep neural networks as scientific models. *Trends in cognitive sciences*, *23*(4), 305–317.

Cong, S., & Zhou, Y. (2023). A review of convolutional neural network architectures and their optimizations. *Artificial Intelligence Review*, *56*(3), 1905–1969.

Croston, J. D. (1972). Forecasting and stock control for intermittent demands. *Journal of the Operational Research Society*, *23*(3), 289–303. doi: https://doi-org.ezproxy2.utwente.nl/10.2307/3007885

Cunningham, P., Cord, M., & Delany, S. J. (2008). Supervised learning. In *Machine Learning Techniques for Multimedia: Case Studies on Organization and Retrieval"* (pp. 21–49). Berlin, Heidelberg: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-75171-7_2

de FSM Russo, R., & Camanho, R. (2015). Criteria in ahp: A systematic review of literature. *Procedia Computer Science*, *55*, 1123–1132.

Doszyń, M. (2022). Biasedness of forecast errors: an intermittent demand perspective. *Procedia Computer Science*, *207*, 644–653.

Fattah, J., Ezzine, L., Aman, Z., Moussami, H. E., & Lachhab, A. (2018). Forecasting of demand using arima model. *International Journal of Engineering Business Management*, *10*, 1847979018808673. doi: https://doi.org/10.1177/1847979018808673

Fildes, R., & Hastings, R. (1994, 01). The organization and improvement of market forecasting. *Journal of the Operational Research Society*, *45*. doi: https://doi.org/10.1057/jors.1994.1

Fildes, R., & Stekler, H. (2002, 02). The state of macroeconomic forecasting. *Journal of Macroeconomics*, *24*, 435-468. doi: https://doi.org/10.1016/S0164-0704(02)00055-1

Fu, T. C. (2011). A review on time series data mining. *Engineering Applications of Artificial Intelligence*, *24*(1), 164–181.

Gardner, E. S. (2006). Exponential smoothing: The state of the art — Part II. *International Journal of Forecasting*, *22*(4), 637-666. doi: https://doi.org/10.1016/j.ijforecast.2006.03.005

Gardner Jr, E. S. (1985). Exponential smoothing: The state of the art. *Journal of forecasting*, *4*(1), 1–28. doi: https://doi.org/10.1002/for.3980040103

Gautam, A., & Singh, V. (2020). Parametric versus non-parametric time series forecasting methods: A review. *Journal of Engineering Science & Technology Review*, *13*(3).

Ghahramani, Z. (2004). Unsupervised learning. In O. Bousquet, U. von Luxburg, & G. Rätsch (Eds.), *Advanced lectures on machine learning* (pp. 72–112). Berlin, Heidelberg: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-28650-9_5

Gilbert, K. (2005). An ARIMA supply chain model. *Management Science*, *51*(2), 305-310. doi: 10.1287/mnsc.1040.0308

Green, K. C., & Armstrong, J. S. (2015). Simple versus complex forecasting: The evidence. *Journal of Business Research*, *68*(8), 1678-1685. (Special Issue on Simple Versus Complex Forecasting) doi: https://doi.org/10.1016/j.jbusres.2015.03.026

Gutierrez, R. S., Solis, A. O., & Mukhopadhyay, S. (2008). Lumpy demand forecasting using neural networks. *International journal of production economics*, *111*(2), 409–420.

Hao, Y., Dong, L., Wei, F., & Xu, K. (2021). Self-attention attribution: Interpreting information interactions inside transformer. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 35, pp. 12963–12971).

Harfeldt-Berg, M., & Olhager, J. (2024). The customer order decoupling point in empirical operations and supply chain management research: a systematic literature review and framework. *International Journal of Production Research*, *62*(17), 6380–6399. doi: https://doi.org/10.1080/00207543.2024.2314164

Heerkens, H., & Van Winden, A. (2021). *Solving managerial problems systematically*. Routledge.

Helgesson, O., & Laszlo, N. (2023). *Sparse time series demand forecasting for intermittent availability* (Unpublished master's thesis). University of Gothenburg.

Herzen, J., Lässig, F., Piazzetta, S. G., Neuer, T., Tafti, L., Raille, G., ... others (2022). Darts: User-friendly modern machine learning for time series. *Journal of Machine Learning Research*, *23*(124), 1–6. Retrieved from `http://jmlr.org/papers/v23/21-1177.html`

Hoffmann, M. A., Lasch, R., & Meinig, J. (2022). Forecasting irregular demand using single hidden layer neural networks. *Logistics Research*, *15*(1), 1–13.

Holt, C. C. (1957). Forecasting seasonals and trends by exponentially weighted moving averages. *International Journal of Forecasting*, *20*(1), 5–10. doi: https://doi.org/10.1016/j.ijforecast.2003.09.015

Huisman Equipment B.V. (n.d.-a). *About huisman.* Retrieved from `https://www.huismanequipment.com/en/about_huisman` (accessed Jul. 09, 2024)

Huisman Equipment B.V. (n.d.-b). *Innovations.* Retrieved from `https://www.werkenbijhuisman.com/innovations` (accessed Jul. 09, 2024)

Hwang, J.-S., Lee, S.-S., Gil, J.-W., & Lee, C.-K. (2024, 07). Determination of Optimal Batch Size of Deep Learning Models with Time Series Data. *Sustainability*, *16*, 5936. doi: 10.3390/su16145936

Hyndman, R. J. (2006). Another look at forecast-accuracy metrics for intermittent demand. *Foresight: The International Journal of Applied Forecasting*, *4*(4), 43–46.

Hyndman, R. J. (2014). Measuring forecast accuracy. *Business forecasting: Practical problems and solutions*, 177–183.

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: principles and practice*. OTexts.

Hyndman, R. J., & Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for r. *Journal of statistical software*, *27*, 1–22.

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, *22*(4), 679-688. doi: https://doi.org/10.1016/j.ijforecast.2006.03.001

Janiesch, C., Zschech, P., & Heinrich, K. (2021). Machine learning and deep learning. *Electronic Markets*, *31*(3), 685–695.

Jeon, Y., & Seong, S. (2022). Robust recurrent network model for intermittent time-series forecasting. *International Journal of Forecasting*, *38*(4), 1415-1425. doi: https://doi.org/10.1016/j.ijforecast.2021.07.004

Karthikeswaren, R., Kayathwal, K., Dhama, G., & Arora, A. (2021). A survey on classical and deep learning based intermittent time series forecasting methods. *Proceedings of the*

International Joint Conference on Neural Networks, *2021-July*. doi: 10.1109/IJCNN52387.2021.9533963

Kiefer, D., Grimm, F., Bauer, M., & Van Dinther, C. (2021). Demand forecasting intermittent and lumpy time series: Comparing statistical, machine learning and deep learning methods. In *Proceedings of the 54th hawaii international conference on system sciences.* doi: http://dx.doi.org/10.24251/HICSS.2021.172

Kim, T., Sharda, S., Zhou, X., & Pendyala, R. M. (2020). A stepwise interpretable machine learning framework using linear regression (LR) and long short-term memory (LSTM): City-wide demand-side prediction of yellow taxi and for-hire vehicle (FHV) service. *Transportation Research Part C: Emerging Technologies*, *120*. doi: https://doi.org/10.1016/j.trc.2020.102786

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. doi: https://doi.org/10.48550/arXiv.1412.6980

Lawrence, M., Goodwin, P., O'Connor, M., & Önkal, D. (2006). Judgmental forecasting: A review of progress over the last 25 years. *International Journal of Forecasting*, *22*(3), 493-518. (Twenty five years of forecasting) doi: https://doi.org/10.1016/j.ijforecast.2006.03.007

Lawrence, M. J., Edmundson, R. H., & O'Connor, M. J. (1986). The accuracy of combining judgemental and statistical forecasts. *Management Science*, *32*(12), 1521–1532. Retrieved from `http://www.jstor.org/stable/2631827`

Leal, J. E. (2020). Ahp-express: A simplified version of the analytical hierarchy process method. *MethodsX*, *7*, 100748. Retrieved from `https://www.sciencedirect.com/science/article/pii/S2215016119303243` doi: https://doi.org/10.1016/j.mex.2019.11.021

Levén, E., & Segerstedt, A. (2004). Inventory control with a modified Croston procedure and erlang distribution. *International Journal of Production Economics*, *90*(3), 361-367. doi: https://doi.org/10.1016/S0925-5273(03)00053-7

Li, Z., Liu, F., Yang, W., Peng, S., & Zhou, J. (2021). A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE transactions on neural networks and learning systems*, *33*(12), 6999–7019.

Liang, J., He, X., Xiao, H., & Wu, C. (2024). Offshore wind power prediction based on two-stage hybrid modeling. *Energy Strategy Reviews*, *54*, 101468.

Lim, B., Arık, S. Ö., Loeff, N., & Pfister, T. (2021). Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, *37*(4), 1748–1764.

Liu, X., & Wang, W. (2024). Deep time series forecasting models: A comprehensive survey. *Mathematics*, *12*(10), 1504.

Lolli, F., Gamberini, R., Regattieri, A., Balugani, E., Gatos, T., & Gucci, S. (2017). Single-hidden layer neural networks for forecasting intermittent demand. *International Journal of Production Economics*, *183*, 116–128.

Luo, X., Yin, W., & Li, Z. (2024). Spatio-temporal graph neural network with hidden confounders for causal forecast. *CEUR Workshop Proceedings*, *3708*, 157 – 169.

Luochen, X., & Hasachoo, N. (2021). The study of irregular demand forecasting for medicines: The case study of abc medical center hospital. In *2021 10th international conference on industrial technology and management (icitm)* (p. 115-120). doi: https://doi.org/10.1109/ICITM52822.2021.00028

Makridakis, S., Chatfield, C., Hibon, M., Lawrence, M., Mills, T., Ord, K., & Simmons, L. F. (1993). The m2-competition: A real-time judgmentally based forecasting study. *International Journal of Forecasting*, *9*(1), 5-22. doi: https://doi.org/10.1016/0169-2070(93)90044-N

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). Statistical and machine learning forecasting methods: Concerns and ways forward. *PloS one*, *13*(3), e0194889.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2020). The m4 competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, *36*(1), 54-74. doi: https://doi.org/10.1016/j.ijforecast.2019.04.014

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2022). M5 accuracy competition: Results, findings, and conclusions. *International Journal of Forecasting*, *38*(4), 1346–1364.

Makridakis, S., Spiliotis, E., Assimakopoulos, V., Semenoglou, A.-A., Mulder, G., & Nikolopoulos, K. (2023). Statistical, machine learning and deep learning forecasting methods: Comparisons and ways forward. *Journal of the Operational Research Society*, *74*(3), 840–859. doi: https://doi.org/10.1080/01605682.2022.2118629

Maleki, N., Lundström, O., Musaddiq, A., Jeansson, J., Olsson, T., & Ahlgren, F. (2024). Future energy insights: Time-series and deep learning models for city load forecasting. *Applied Energy*, *374*, 124067.

Mehtab, S., & Sen, J. (2022). Analysis and forecasting of financial time series using cnn and lstm-based deep learning models. In *Advances in distributed computing and machine learning: Proceedings of icadcml 2021* (pp. 405–423).

Mirshahi, S., Brandtner, P., & Oplatková, Z. K. (2024). Intermittent time series demand forecasting using dual convolutional neural networks. In *Mendel* (Vol. 30, pp. 51–59).

Mishra, R., Kumar, R., Dhingra, S., Sengupta, S., Sharma, T., & Gautam, G. D. (2022). Adaptive grey model (AGM) approach for judgemental forecasting in short-term manufacturing demand. *Materials Today: Proceedings*, *56*, 3740–3746.

Montero-Manso, P., & Hyndman, R. J. (2021). Principles and algorithms for forecasting groups of time series: Locality and globality. *International Journal of Forecasting*, *37*(4), 1632–1653.

Montgomery, D. C., Johnson, L. A., & Gardiner, J. S. (1990). Forecasting and time series analysis. *n.d.*.

Muhaimin, A., Prastyo, D. D., & Lu, H. H.-S. (2021). Forecasting with recurrent neural network in intermittent demand data. In *2021 11th international conference on cloud computing, data science & engineering (confluence)* (pp. 802–809).

Mukhopadhyay, S., Solis, A. O., & Gutierrez, R. S. (2012). The accuracy of non-traditional versus traditional methods of forecasting lumpy demand. *Journal of Forecasting*, *31*(8), 721–735. doi: https://doi.org/10.1002/for.1242

Nasteski, V. (2017). An overview of the supervised machine learning methods. *Horizons. b*, *4*(51-62), 56.

National Institute of Standards and Technology. (n.d.). *Univariate time series models.* Retrieved from `https://www.itl.nist.gov/div898/handbook/pmc/section4/pmc44.htm` ((accessed Jul. 09, 2024))

Nguyen, T. K. T., Antoshchuk, S., Nikolenko, A., Tran, K. T., & Babilunha, O. (2019). Non-stationary time series prediction using one-dimensional convolutional neural network models. *Herald of Advanced Information Technology*, *1*(3), 362–372.

Olhager, J. (2010). The role of the customer order decoupling point in production and supply chain management. *Comput. Ind.*, *61*, 863-868.

Ostertagová, E., & Ostertag, O. (2011). The simple exponential smoothing model. In *The 4th international conference on modelling of mechanical and mechatronic systems, technical university of košice, slovak republic, proceedings of conference* (pp. 380–384).

Paik, J., Baek, S.-J., Kim, J.-W., & Ko, K. (2023). Influence of social overhead capital facilities on housing prices using machine learning. *Applied Sciences*, *13*(19). doi: 10.3390/app131910732

Park, S., & Kwak, N. (2017). Analysis on the dropout effect in convolutional neural networks. In S.-H. Lai, V. Lepetit, K. Nishino, & Y. Sato (Eds.), *Computer vision – accv 2016* (pp. 189–204). Cham: Springer International Publishing. doi: https://doi.org/10.1007/978-3-319-54184-6_12

Pelletier, C., Webb, G. I., & Petitjean, F. (2019). Temporal convolutional neural network for

the classification of satellite image time series. *Remote Sensing*, *11*(5), 523.

Petropoulos, F., Apiletti, D., Assimakopoulos, V., Babai, M. Z., Barrow, D. K., Ben Taieb, S., ... Ziel, F. (2022). Forecasting: theory and practice. *International Journal of Forecasting*, *38*(3), 705-871. doi: https://doi.org/10.1016/j.ijforecast.2021.11.001

Ravinder, H. V. (2013). Forecasting With Exponential Smoothing - What's the right smoothing constant? *The Review of Business Information Systems (Online)*, *17*(3), 117. doi: http://dx.doi.org/10.19030/rbis.v17i3.8001

Refaeilzadeh, P., Tang, L., & Liu, H. (2009). Cross-validation. In L. LIU & M. T. ÖZSU (Eds.), *Encyclopedia of database systems* (pp. 532–538). Boston, MA: Springer US. doi: https://doi.org/10.1007/978-0-387-39940-9_565

Reimers, N., & Gurevych, I. (2017). Optimal Hyperparameters for Deep LSTM-Networks for Sequence Labeling Tasks. *arXiv preprint arXiv:1707.06799*. doi: https://doi.org/10.48550/arXiv.1707.06799

Rosienkiewicz, M. (2013). Artificial intelligence methods in spare parts demand forecasting. *Logistics and Transport*, *18*.

Saaty, T. L. (1977). A scaling method for priorities in hierarchical structures. *Journal of mathematical psychology*, *15*(3), 234–281.

Saaty, T. L. (2008). Decision making with the analytic hierarchy process. *International journal of services sciences*, *1*(1), 83–98.

Şahin, M., Kizilaslan, R., & Demirel, Ö. F. (2013). Forecasting aviation spare parts demand using croston based methods and artificial neural networks. *Journal of Economic and Social Research*, *15*(2), 1.

Schmidhuber, J., & Hochreiter, S. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Schultz, C. R. (1987). Forecasting and inventory control for sporadic demand under periodic review. *Journal of the Operational Research Society*, *38*(5), 453–458. doi: https://doi.org/10.1057/jors.1987.74

Shafi, I., Sohail, A., Ahmad, J., Espinosa, J. C. M., López, L. A. D., Thompson, E. B., & Ashraf, I. (2023). Spare parts forecasting and lumpiness classification using neural network model and its impact on aviation safety. *Applied Sciences*, *13*(9), 5475.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Syntetos, A. (2001). Forecasting of intermittent demand. *Brunel University Uxbridge*.

Syntetos, A. A., & Boylan, J. E. (2001). On the bias of intermittent demand estimates. *International journal of production economics*, *71*(1-3), 457–466.

Syntetos, A. A., Boylan, J. E., & Croston, J. (2005). On the categorization of demand patterns. *Journal of the operational research society*, *56*(5), 495–503.

Sze, V., Chen, Y.-H., Yang, T.-J., & Emer, J. S. (2017). Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE*, *105*(12), 2295–2329.

Taherdoost, H., & Madanchian, M. (2023). Multi-criteria decision making (mcdm) methods and concepts. *Encyclopedia*, *3*(1), 77–87.

Taib, S. A. T., Abu, N., Senawi, A., & Go, C. K. (2025). The implementation of long-short term memory for tourism industry in malaysia. *Journal of Advanced Research in Applied Sciences and Engineering Technology*, *46*(2), 90–97.

Tashman, L. J. (2000). Out-of-sample tests of forecasting accuracy: an analysis and review. *International Journal of Forecasting*, *16*(4), 437-450. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0169207000000650` (The M3- Competition) doi: https://doi.org/10.1016/S0169-2070(00)00065-0

Tavana, M., Soltanifar, M., & Santos-Arteaga, F. J. (2023). Analytical hierarchy process: Revolution and evolution. *Annals of operations research*, *326*(2), 879–907.

van Donk, D. P., & van Doorne, R. (2016). The impact of the customer order decoupling point on type and level of supply chain integration. *International Journal of Production Research*, *54*(9), 2572–2584.

Vankadara, L. C., Faller, P. M., Hardt, M., Minorics, L., Ghoshdastidar, D., & Janzing, D. (2022, August). Causal forecasting: generalization bounds for autoregressive models. In *Uncertainty in artificial intelligence* (pp. 2002–2012).

Varma, S., & Simon, R. (2006). Bias in error estimation when using cross-validation for model selection. *BMC bioinformatics*, *7*, 1–8.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*.

Watanabe, S. (2023). Tree-structured parzen estimator: Understanding its algorithm components and their roles for better empirical performance. *arXiv preprint arXiv:2304.11127*. doi: https://doi.org/10.48550/arXiv.2304.11127

Wen, Q., Zhou, T., Zhang, C., Chen, W., Ma, Z., Yan, J., & Sun, L. (2022). Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*.

Willemain, T. R., Smart, C. N., Shockor, J. H., & DeSautels, P. A. (1994). Forecasting intermittent demand in manufacturing: a comparative evaluation of Croston's method. *International Journal of Forecasting*, *10*(4), 529-538. doi: https://doi.org/10.1016/0169-2070(94)90021-3

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, *1*(1), 67–82. doi: https://doi.org/10.1109/4235.585893

Wu, H., & Levinson, D. (2021). The ensemble approach to forecasting: A review and synthesis. *Transportation Research Part C: Emerging Technologies*, *132*, 103357.

Zellner, M., Abbas, A. E., Budescu, D. V., & Galstyan, A. (2021). A survey of human judgement and quantitative forecasting methods. *Royal Society open science*, *8*(2), 201187. doi: https://doi.org/10.1098/rsos.201187

Zhang, G. P., Xia, Y., & Xie, M. (2023). Intermittent demand forecasting with transformer neural networks. *Annals of Operations Research*, 1–22.

Zhu, Q., Han, J., Chai, K., & Zhao, C. (2023). Time series analysis based on informer algorithms: A survey. *Symmetry*, *15*(4), 951.

Zhu, X., Ninh, A., Zhao, H., & Liu, Z. (2021). Demand forecasting with supply-chain information and machine learning: Evidence in the pharmaceutical industry. *Production and Operations Management*, *30*(9), 3231 – 3252. doi: https://doi.org/10.1111/poms.13426

Appendix A

AHP-express ranking

A.1 Scoring Scale

TABLE A.1: Scale of relative importance for pairwise comparison in AHP (Saaty, 1977)

Intensity of importance	Definition	Explanation
1	Equal importance	Two alternatives contribute equally to the objective
3	Moderate importance	Experience and judgement slightly favour one alternative over the other
5	Strong importance	Experience and judgement strongly favour one alternative over the other
7	Very strong or demonstrated importance	An alternative is favoured very strongly over another; its dominance is demonstrated in practice
9	Extreme importance	The evidence favouring one alternative over another is of the highest possible order of affirmation
2, 4, 6, 8	Intermediate values	These intensities of importance are used when compromise is needed
Reciprocals	Values for inverse comparison	If alternative i has one of the above nonzero numbers assigned to it when compared with alternative j, then j has the reciprocal value when compared with i

A.2 Scoring Methodology

We use the nomenclature defined in Table A.2 for the alternative purchase groups.

Symbol	Purchase group
A_1	Bearings
A_2	Gearboxes
A_3	Hooks
A_4	Winches
A_5	Sheaves

TABLE A.2: Nomenclature of the alternatives in AHP-express

The priority vectors $\mathbf{PSC}_i$ for all $i \in \{1, 2, \ldots, nc\}$ and $\mathbf{PC}$ are calculated based on expert judgments provided by the decision-maker, the supply chain manager of Huisman Equipment B.V., using the 9-point scale depicted in Table A.1 to assign relative preferences to sub-criteria and criteria, respectively. Conversely, the priority matrices $\mathbf{PASC}_i$ for all $i \in \{1, 2, \ldots, nc\}$ calculate their priorities using both quantitative data and expert judgment to assign relative preferences. To assign these relative preferences of the alternatives for the sub-criteria within the criteria data, procurement value, production criticality, and operational criticality, we use the following approaches:

- **Data:** The data quality of the alternatives is assessed by comparing their percentage of transactions with a dummy part code and their percentage of transactions with an invalid (i.e., empty or ≤ 0) quantity and unit purchase price. The empirical absolute Complementary Cumulative Distribution Function (CCDF) in Figure A.1 evaluates the data volume for the alternatives.
- **Procurement value:** For the SKU procurement value sub-criterion, we use the box plots in Figure A.2 to compare the average purchase price per transaction of the alternatives. The secondary y-axis in the figure shows the total procurement costs for each alternative over the past five years, which is used to compared the alternatives based on the total grouped procurement value sub-criterion.

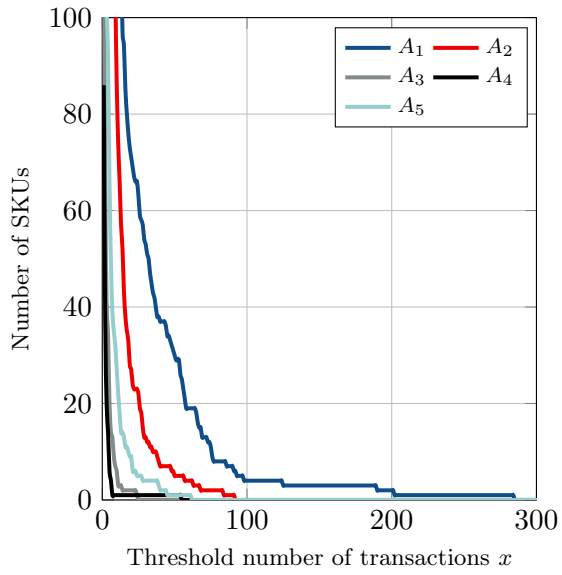


FIGURE A.1: Empirical absolute CCDF of the number of SKUs with more than x transactions for each alternative purchase group

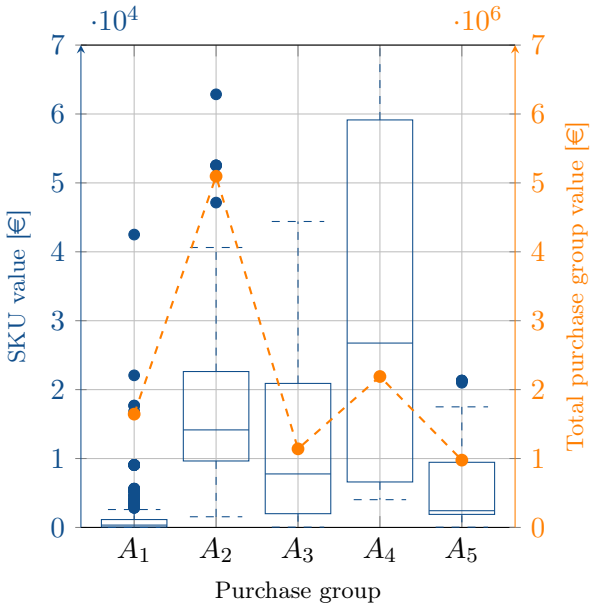


FIGURE A.2: Box plots of the average SKU procurement prices per transaction and total procurement costs for each alternative purchase group over the past five years

- **Production criticality:** For the critical path sub-criterion, we use expert opinion of the decision-maker to compare the extent to which each alternative is generally located in the critical path of a project. We use procurement lead time data in Figure A.3 to compare the alternatives for the lead time sub-criterion, as long procurement lead time cause operational challenges for Huisman projects.

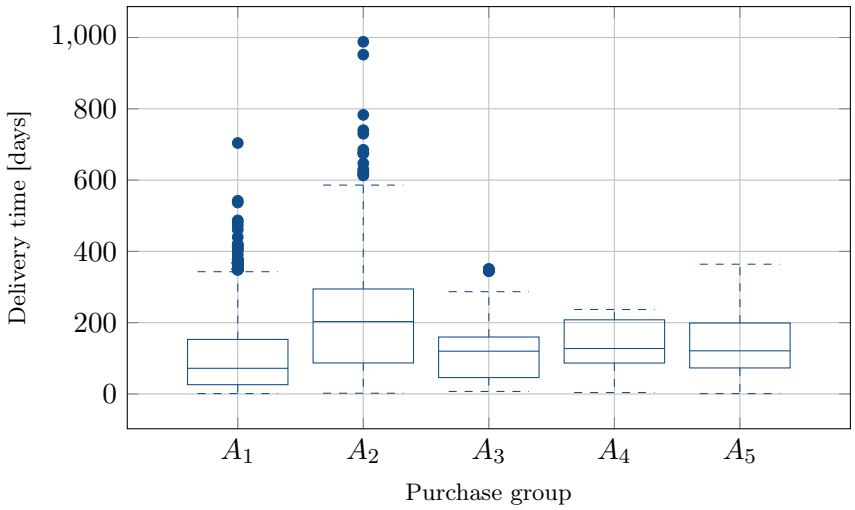


FIGURE A.3: Distribution of individual replenishment lead times for all purchase orders, grouped by alternative purchase group

A.3 Assigned Preferences and Priority Calculations

All resulting relative preferences of the alternatives for each sub-criterion are depicted in Table A.3. Based on expert judgement, we also assign the relative priorities of the sub-criteria and criteria in Table A.4 and A.5, respectively.

$SC_{1,1}$	A_1	A_2	A_3	A_4	A_5
A_4	5	4	2	1	3
$pasc^1_{a,1}$	.09	.11	.22	.44	.15

(A) Data quality sub-criterion

$SC_{1,2}$	A_1	A_2	A_3	A_4	A_5
A_1	1	7	9	7	4
$pasc^1_{a,2}$	.61	.09	.07	.09	.15

(B) Data volume sub-criterion

$SC_{2,1}$	A_1	A_2	A_3	A_4	A_5
A_4	7	3	5	1	5
$pasc^2_{a,1}$	.08	.18	.11	.53	.11

(C) SKU value sub-criterion

$SC_{2,2}$	A_1	A_2	A_3	A_4	A_5
A_2	3	1	7	2	4
$pasc^2_{a,2}$	.15	.45	.06	.22	.11

(D) Purchase group value sub-criterion

$SC_{3,1}$	A_1	A_2	A_3	A_4	A_5
A_1	1	1	2	5	7
$pasc^3_{a,1}$	.35	.35	.18	.07	.05

(E) Critical path sub-criterion

$SC_{3,2}$	A_1	A_2	A_3	A_4	A_5
A_2	4	1	4	3	3
$pasc^3_{a,2}$	.12	.46	.12	.15	.15

(F) Lead time sub-criterion

TABLE A.3: Priority calculation of the alternatives for the sub-criteria

C_1	$SC_{1,1}$	$SC_{1,2}$
$SC_{1,2}$	2	1
cg_j^1	.33	.67

(A) Data criterion

C_2	$SC_{2,1}$	$SC_{2,2}$
$SC_{2,1}$	1	5
cg_j^2	.83	.17

(B) Procurement value criterion

C_3	$SC_{3,1}$	$SC_{3,2}$
$SC_{3,1}$	1	3
cg_j^3	.75	.25

(C) Production criticality criterion

TABLE A.4: Priority calculation of the sub-criteria within each criterion

	C_1	C_2	C_3
C_1	1	5	1
pc_i	.45	.09	.45

TABLE A.5: Priority calculation of the criteria

A.4 Final Priority Calculations and Alternative Ranking

After assigning all relative preferences, we calculate the relative priorities using Equation (2.1). The relative priorities are used to construct matrices **MPSC** and **MPASC**, and we multiply the two to obtain matrix **PAC** using Equations (2.2)-(2.4). We obtain the following matrices:

$$\mathbf{MPSC} = \begin{bmatrix} 0.333 & 0.667 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.833 & 0.167 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.750 & 0.250 \end{bmatrix} \tag{A.1}$$

$$\mathbf{MPASC} = \begin{bmatrix} 0.088 & 0.110 & 0.219 & 0.438 & 0.146 \\ 0.607 & 0.0870 & 0.067 & 0.087 & 0.152 \\ 0.076 & 0.178 & 0.107 & 0.533 & 0.107 \\ 0.150 & 0.449 & 0.064 & 0.225 & 0.112 \\ 0.352 & 0.352 & 0.176 & 0.070 & 0.050 \\ 0.115 & 0.462 & 0.115 & 0.154 & 0.154 \end{bmatrix} \tag{A.2}$$

$$\mathbf{PAC} = \mathbf{MPSC} \cdot \mathbf{MPASC} = \begin{bmatrix} 0.434 & 0.094 & 0.118 & 0.204 & 0.150 \\ 0.088 & 0.223 & 0.100 & 0.482 & 0.108 \\ 0.293 & 0.379 & 0.161 & 0.091 & 0.076 \end{bmatrix} \tag{A.3}$$

Last, we multiply the 3-dimensional row vector **PC** with 3×5 matrix **PAC**, i.e., we calculate the weighted sum of the priorities of the alternatives for each criterion (given in the columns of the **PAC** matrix) using the priorities of the criteria in the **PC** vector as weights:

$$\mathbf{PC} = \begin{bmatrix} 0.455 & 0.091 & 0.455 \end{bmatrix} \tag{A.4}$$

$$\mathbf{PA} = \mathbf{PC} \cdot \mathbf{PAC} = \begin{bmatrix} 0.338 & 0.234 & 0.136 & 0.178 & 0.113 \end{bmatrix} \tag{A.5}$$

Appendix B

Non-zero Monthly Demand Distribution for Bearings

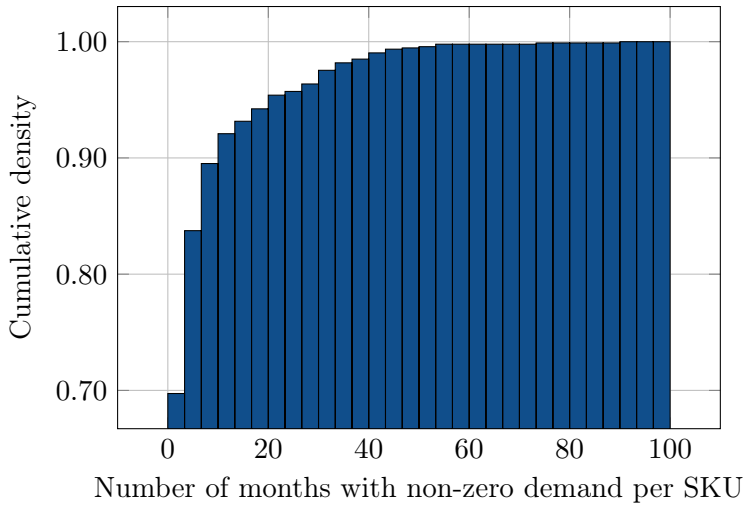
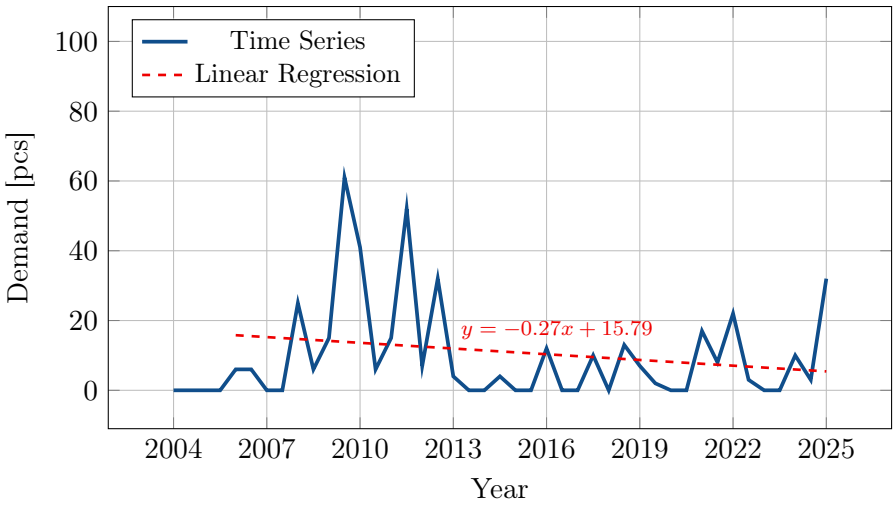


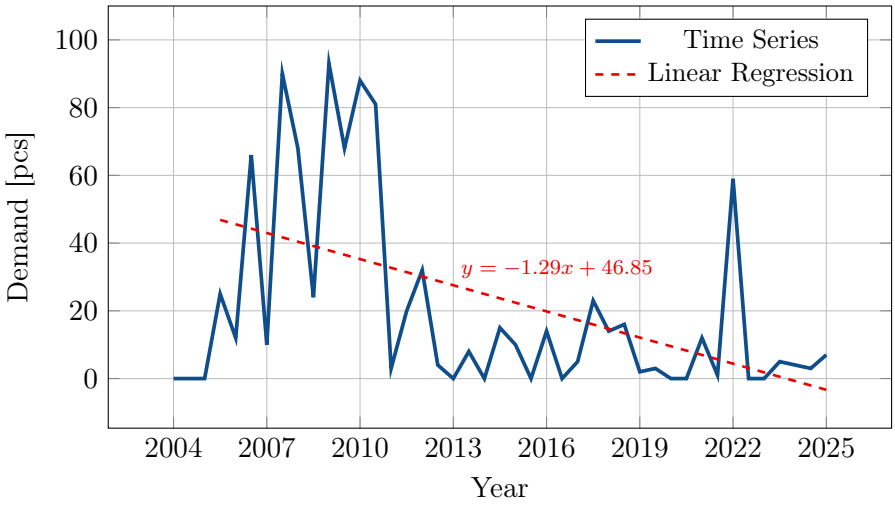
FIGURE B.1: Empirical cumulative distribution of the number of months with non-zero demand per SKU in the bearings purchase group from 2004 to 2024

Appendix C

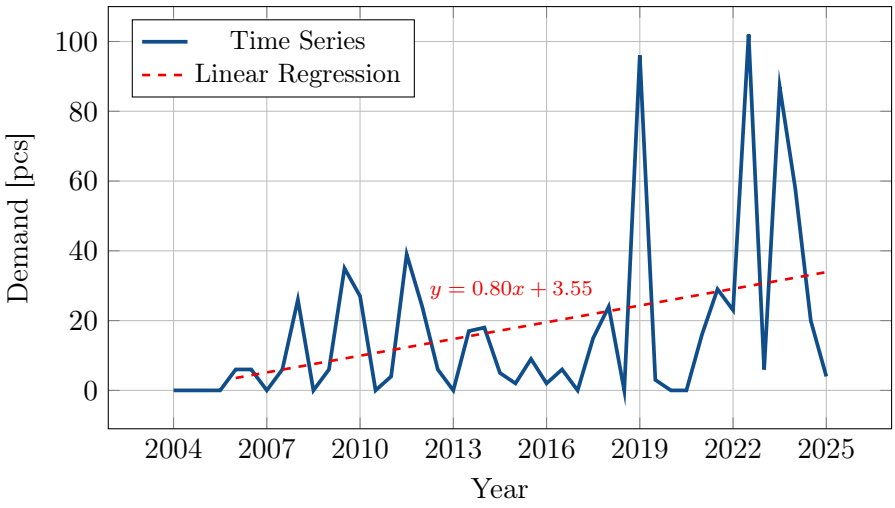
Fitted Regression Lines from Linear Regression t-test



(A) Fitted linear regression trends for LLI-1



(B) Fitted linear regression trends for LLI-2



(c) Fitted linear regression trends for LLI-3

FIGURE C.1: Fitted linear regression trends for the historical demand time series of the selected LLIs, used to assess the presence and direction of long-term demand trends.

Appendix D

Temporal Encodings for Multivariate Forecasts

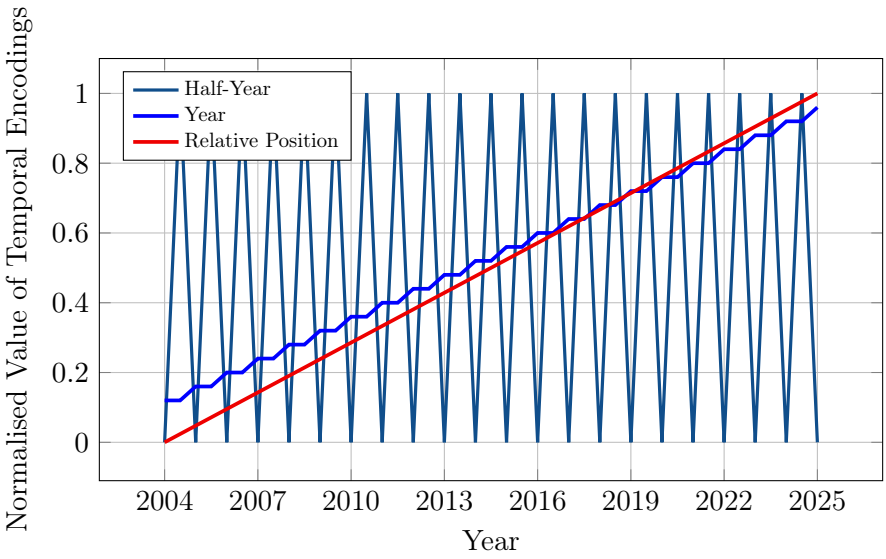


FIGURE D.1: Graphical representation of the normalised temporal encodings for multivariate forecasting

Appendix E

Project Classification Scheme for Project-related Covariates

TABLE E.1: Project classification scheme used to structure project-related covariates by equipment type in the multivariate forecasting models.

Project Classification Scheme			
ClassID	Description	ClassID	Description
1	Pedestal Mounted Crane	8	PTC/Ringer Crane
2	Heavy Lift Mast Crane	9	Automated Stacking Crane
3	Offshore Mast Crane	10	Overhead/Gantry Crane
4	Knuckle Boom Crane	11	Pile Gripper
5	Tub Mounted Crane	12	Drilling Equipment
6	Leg Encircling Crane	13	Pipelay Equipment
7	Hybrid Boom Crane	14	Other