Computers & Industrial Engineering 211 (2026) 111645

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Optimizing maintenance of heavy equipment: A data-driven approach

Lais Wehbi [a,*], Sandjai Bhulai [a], Jesper Slik [b]

[a] Vrije Universiteit Amsterdam, De Boelelaan 1111, Amsterdam, 1081 HV, The Netherlands
[b] Pon, Stadionplein 28, Amsterdam, 1076 CM, The Netherlands

ARTICLE INFO

Keywords:
Heavy equipment maintenance
Priority scoring
Predictive analytics
Multi-Depot Capacitated Vehicle Routing Problem
Self-Organizing Maps

ABSTRACT

Timely maintenance of heavy equipment is crucial in construction yet hindered by limited adoption of data-driven methods. We introduce an integrated framework that links priority scoring, predictive analytics, and routing optimization. Our system achieves a 64.4% reduction in prediction error versus a naïve current-SMU baseline on a ten-machine evaluation set without specialized monitoring. Our machine learning models attain MAPE = 4.24% (labor cost) and 9.32% (service duration), substantially outperforming company baselines. We model routing as a Multi-Depot Capacitated Vehicle Routing Problem (MDCVRP) and solve it with a modified Cluster-First, Route-Second approach combining ALNS, 2-OPT*, and a novel SOM-based adaptation. Across varying instances, results show a 41.93% reduction in total time and a 43.86% improvement in high-priority route position relative to current practice. Sensitivity analyses (priority weights, clustering, travel-time uncertainty) confirm robustness. The framework supports better availability and cost savings, and can be extended to multi-equipment and stochastic, real-time settings.

1. Introduction

The construction industry faces increasing equipment maintenance challenges amid urbanization and infrastructure expansion. A (United Nations Environment Programme, 2019) report predicts that the global building stock will double by 2050, escalating demand for maintaining construction machinery, including excavators, bulldozers, cranes, loaders, and dump trucks. Generating 15% of global GDP and employing 10% of the workforce (Hilti Corporation, 2023), the sector is shifting toward sustainability and safety, making effective maintenance essential for operational efficiency, cost reduction, and environmental impact reduction (Amrina & Aridharma, 2016).

However, maintenance is hindered by several factors including site mobility, diverse equipment needs, and seasonal work patterns (Abioye et al., 2021; Niu et al., 2016, 2017). These challenges lead to poor maintenance practices that increase costs, reduce productivity, and delay projects (Close & Tideswell, 2012). From a financial perspective, maintenance represents 20%–40% of operating costs and drives 30%–50% of equipment effectiveness losses (Armstrong et al., 2024). Furthermore, unplanned downtime costs up to $260 per hour per machine (Hicks, 2019), with industry downtime rates reaching 20%–30%, leading to multi-million-dollar annual losses (Hill, 2021).

Beyond economic impacts, environmental considerations are equally consequential. The construction sector accounts for 37% of global CO_2 emissions and 132 exajoules of energy consumption (UN Environment Programme, 2024). Maintenance issues exacerbate this problem by increasing fuel use, emissions, and waste, whereas proper maintenance can substantially reduce fuel consumption by 6.6% and emissions by 20% (Aune et al., 2020; Hong & Lü, 2022). Additionally, well-maintained equipment lasts 50%–60% longer, reducing premature replacements (Green, 2023).

Despite these clear benefits, the industry continues to lag in digitization, with only 1% annual labor productivity growth, below the 2.8% global average and 3.6% for manufacturing (Hovnanian et al., 2019). This digital gap stems from several barriers including fragmented data, lack of standardization, resistance to change, high implementation costs, and regulatory constraints (Sivanuja et al., 2024), with 72% of firms still relying on paper-based processes (Bluebeam Inc., 2024) and poor data management costing the industry $1.85 trillion globally in 2020 (Autodesk Inc. & FMI Corporation, 2021).

While construction companies typically invest only 1%–2% of revenue in technology (Blanco et al., 2023), organizations achieving digital transformation report 14%–15% productivity gains (Koeleman et al., 2019). Nevertheless, the potential benefits are substantial data-driven approaches could reduce maintenance costs by 5%–10% and increase uptime by 10%–20% (Coleman et al., 2017). Moreover, analytics support better project decision-making in critical areas such as bidding, subcontractor evaluation, and risk management (Hovnanian et al., 2019).

* Corresponding author.
E-mail address: l.wehbi@vu.nl (L. Wehbi).

https://doi.org/10.1016/j.cie.2025.111645
Received 18 March 2025; Received in revised form 14 August 2025; Accepted 27 October 2025
Available online 30 October 2025

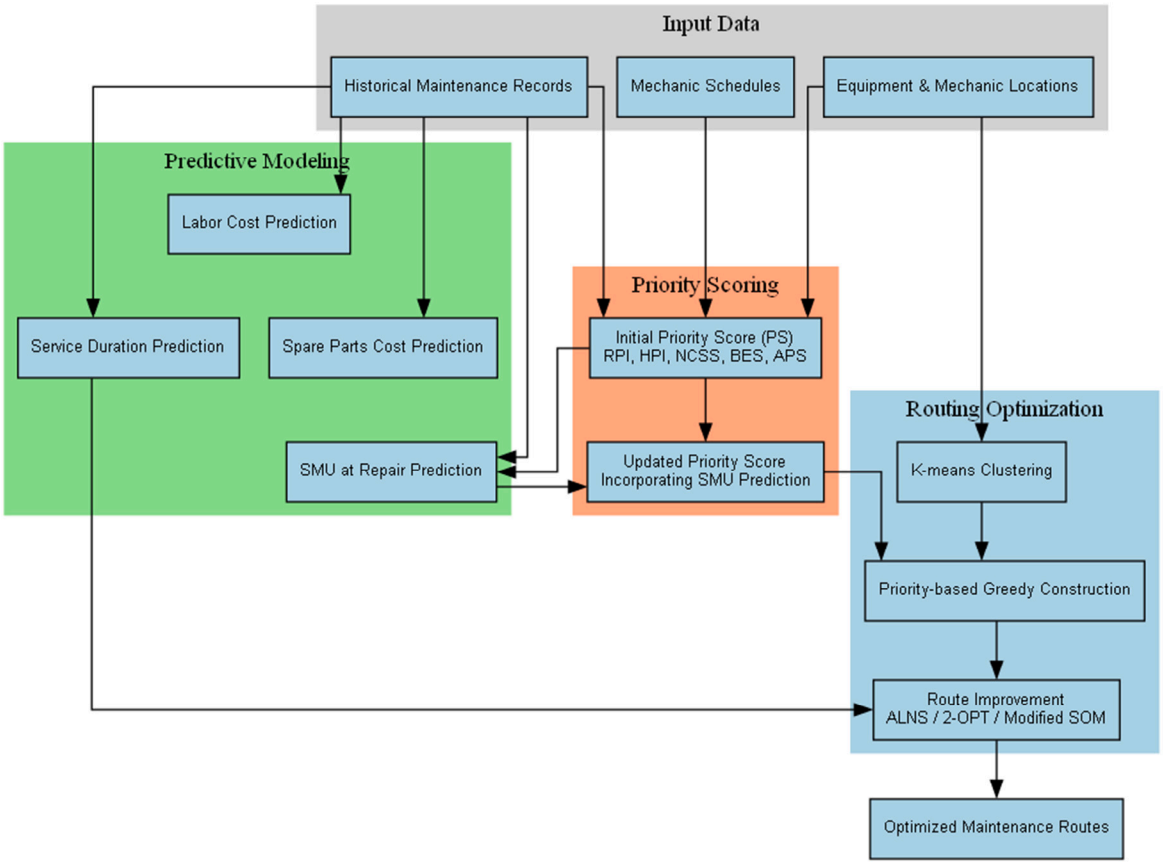

Fig. 1. Conceptual Framework: Integrated approach for heavy equipment maintenance optimization.

To address these challenges, we propose a maintenance optimization framework that integrates three components into a unified data processing pipeline (Fig. 1): (1) a predictive modeling system for maintenance variables, (2) a priority scoring mechanism for service urgency quantification, and (3) a routing optimization algorithm for resource allocation. Unlike traditional approaches that treat these elements as isolated processes, our framework establishes relationships between components, allowing for data-driven optimization for improved resource allocation efficiency.

The framework operates through a sequential yet interconnected workflow. First, the Predictive Modeling component forecasts key maintenance variables (labor costs, service durations, spare parts costs, and Service Meter Unit readings). These predictions, particularly SMU forecasts directly feed into the Priority Scoring component, which quantifies service urgency using multiple operational indicators, as well as incorporating predicted SMU values for a future outlook on prioritization. These scores then serve as weighted inputs to the Routing Optimization component, which employs k-means clustering and priority-based greedy construction to generate initial routes, subsequently refined using advanced heuristic techniques including a novel adaptation of self organizing maps (SOM). Our key contributions include:

1. A flexible priority scoring system adaptable to business objectives, eliminating the need for machine health metrics.
2. Accurate predictive models for key maintenance variables, integrating SMU forecasts into priority scoring.
3. A MILP formulation of MDCVRP integrating priority-based routing with heuristic techniques.
4. To the best of our knowledge, this represents one of the first adaptations of Self-Organizing Maps (SOM) to MDCVRP in maintenance routing.
5. Validation with real-world data, demonstrating performance improvements over industry practices.

The paper is structured as follows: Section 2 reviews related work. Section 3 details the priority scoring system. Section 4 discusses predictive modeling. Section 5 presents the routing approach. Section 6 explores practical implications, limitations, and future research. Section 7 concludes the study.

2. Literature review

This section reviews recent advancements in predictive maintenance, prioritization strategies, and vehicle routing problems in maintenance scheduling. By identifying research gaps, we establish the foundation for our integrated approach.

2.1. Data-driven approaches and predictive maintenance

The construction industry is gradually adopting data-driven maintenance to replace traditional fixed-interval schedules. Abioye et al. (2021) emphasize AI's potential to improve reliability and efficiency, though adoption remains slow due to fragmented data, lack of standardization, and organizational resistance (Chen & Ying, 2022). IoT and big data analytics have shown promise, as demonstrated by Him et al. (2019), who developed an IoT-based predictive maintenance system that reduced downtime. Predictive modeling enhances maintenance efficiency, with Cheng et al. (2020) integrating BIM and IoT for real-time condition monitoring.

Zhao et al. (2022) introduced a deep learning method for estimating the remaining useful life (RUL) of equipment, outperforming RNN and LSTM. Wang et al. (2024) further advanced this approach by developing multi-objective predictive maintenance models that integrate RUL

prediction with maintenance scheduling, using a deep learning ensemble (CNN-BiLSTM) and mixed-integer linear programming to achieve over 40% reduction in both maintenance time and costs. Digital twins further advance predictive maintenance, with Joe Opoku and Zeeshan (2021) highlighting their ability to create virtual representations of assets for optimized interventions, though challenges remain in data integration and model accuracy.

2.2. Vehicle routing problems in maintenance

Optimizing maintenance technician routing is often modeled as a Multi-Depot Capacitated Vehicle Routing Problem (MDCVRP). Montoya-Torres et al. (2015) reviewed MDCVRP variants, while Harikrishnakumar et al. (2020) applied quantum annealing to improve computational efficiency. López-Santana et al. (2016) developed a Combined Maintenance and Routing (CMR) model integrating preventive maintenance with routing optimization.

Machine learning further improves maintenance routing, as seen in Biteus and Lindgren (2017), which linked predictive modeling with operational planning. Zhang et al. (2020) employed multi-agent reinforcement learning for time-sensitive routing. López-Santana et al. (2023) combined preventive maintenance with technician routing optimization, balancing costs and failure probabilities. Toru and Yilmaz (2023) formulated an MDVRPTW model optimizing maintenance service schedules, while Jbili et al. (2018) integrated vehicle routing with maintenance planning for long-distance transportation.

Table 1 synthesizes the maintenance optimization literature, showing that existing approaches address components in isolation or achieve only partial integration. The predictive maintenance stream, as observed by Cheng et al. (2020) and Zhao et al. (2022), focuses on equipment monitoring and failure prediction but neglects the spatial routing challenges inherent in distributed maintenance operations. Conversely, the routing optimization stream, including Toru and Yilmaz (2023) and Jbili et al. (2018), treats maintenance requirements as static inputs rather than dynamically predicted variables. While Wang et al. (2024) successfully combines RUL prediction with maintenance scheduling for aircraft engines, their approach addresses only depot assignment without explicit routing sequences between service locations. Similarly, López-Santana et al. (2023) integrates routing with predictive maintenance planning but embeds priority merely as an opportunity-cost term rather than developing a comprehensive scoring system, and relies on synthetic data for validation.

Our study bridges these gaps by developing an integrated framework that combines multi-factor priority scoring, predictive analytics for multiple maintenance variables and routing optimization through a novel SOM adaptation to MDCVRP. Unlike approaches requiring specialized IoT infrastructure, our method leverages readily available operational data, making it accessible to typical construction firms while allowing for fast optimization for urgent scheduling needs. This integration, validated with actual construction company data, represents a theoretical contribution and a practical solution for an industry that has historically lagged in digital transformation. Our integrated approach, detailed in the following sections, operationalizes this framework through specific methodological innovations. With this research landscape established, we now present our priority scoring framework that forms the foundation of our integrated approach.

3. Priority scoring framework

This section introduces a priority scoring system that incorporates multiple factors beyond traditional metrics. We first review current industry practices and then present our new prioritization approach.

3.1. Maintenance practices in heavy construction equipment

Maintenance in the construction industry is typically scheduled based on Service Meter Unit (SMU) readings, with preventive maintenance performed at 500 SMU intervals based on manufacturer recommendations and industry experience. Standard metrics used to track adherence include:

- **Relative Pass (RP)**: A binary metric indicating whether maintenance occurs within ±50 SMU of the expected relative interval.
- **Absolute Pass (AP)**: A binary metric indicating whether maintenance aligns with specific SMU milestones within a ±50 SMU tolerance.
- **Best Case Pass (BCP)**: Assigned 1 if either RP or AP is achieved, 0 otherwise.

While these binary metrics are easy to implement, they lack nuance, failing to account for machine criticality, operational constraints, or predictive insights.

3.2. Priority score formulation

To address these limitations, we propose a continuous priority score incorporating multiple operational factors. To ensure balanced contribution from all components despite their different natural scales, we apply min–max normalization to each component before calculating priority scores. For each component $X \in \{\text{RPI, HPI, APS, PMU}\}$, we compute the normalization using only the training window or current scheduling horizon to avoid data leakage:

$$\tilde{X}_m = \begin{cases} \frac{X_m - \min_{training} X}{\max_{training} X - \min_{training} X} & \text{if } \max_{training} X \neq \min_{training} X \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Note that when all values in a component are identical (e.g., all machines have the same score), we set the normalized value to 0 to avoid division by zero. The NCSS and BES components are already scaled to [0,1] through their definitions and are thus excluded from this additional normalization step. The normalized priority score is then calculated as:

$$\text{PS}_m = W_1 \cdot \tilde{\text{RPI}}_m + W_2 \cdot \tilde{\text{HPI}}_m + W_3 \cdot \text{NCSS}_m + W_4 \cdot \tilde{\text{APS}}_m + W_5 \cdot \text{BES}_m, \quad (2)$$

where PS_m is the priority score for machine m, and the weights W_i are constrained by $\sum_{i=1}^{5} W_i = 1$. Each component of this formula is designed to capture a specific aspect of maintenance prioritization.

3.2.1. Recent and historical performance components

$$\text{RPI}_m = \max(0.8 - \text{Recent PM accuracy ratio}_m, 0). \quad (3)$$

The Recent PM Impact (RPI) component provides a short-term view of the machine's recent maintenance performance, where the Recent PM accuracy ratio is calculated as the proportion of the last 3 preventive maintenance services that met the pass criteria (within ±50 SMU of target). Using a continuous scale rather than binary pass/fail metrics, this component can differentiate between machines with varying degrees of maintenance timeliness. The threshold of 0.8 represents an 80% accuracy target, aligning with common industry standards while allowing for some flexibility.

$$\text{HPI}_m = \max(0.8 - \text{Historical PM accuracy ratio}_m, 0). \quad (4)$$

Similarly, the Historical PM Impact (HPI) quantifies long-term maintenance compliance, where the Historical PM accuracy ratio represents the proportion of all preventive maintenance services in the past 12 months that met the pass criteria. The parallel structure between RPI and HPI allows the model to balance immediate and long-term maintenance patterns, both calibrated to the same 80% accuracy threshold.

Table 1
Comparison of maintenance optimization approaches in the literature.

Study	Core components			Implementation features			Integration	Domain
	Priority	Prediction	Routing	IoT-free	Real-time	Real data	level	
Cheng et al. (2020)	×	✓	×	×	✓	✓	Low	Building
Zhao et al. (2022)	×	✓	×	✓	×	✓	Low	Equipment
Wang et al. (2024)	Partial[d]	✓	Partial[a]	✓	×	✓	Medium	Aircraft
da Silva et al. (2021)	✓	×	×	✓	×	✓	Low	General
López-Santana et al. (2016)	×	×	✓	✓	×	×	Low	General
López-Santana et al. (2023)	Partial[b]	✓	✓	✓	×	×	Medium	General
Toru and Yilmaz (2023)	×	×	✓	✓	×	×	Low	Service
Jbili et al. (2018)	×	×	✓	✓	×	Partial[c]	Low	Transport
This study	✓	✓	✓	✓	×	✓	**Full**	**Construction**

[a] "Partial" indicates depot assignment but no explicit travel-routing sequence.
[b] Priority embedded as opportunity-cost term, not an independent scoring system.
[c] Validation uses stylized demand calibrated to a single operator's logbook.
[d] Priority level generated for each engine but not coupled to a dynamic queueing model.

3.2.2. Customer satisfaction integration

$$\mathrm{NCSS}_m = \frac{\mathrm{maxCS} - \text{Average customer satisfaction score}_m}{\mathrm{maxCS} - \mathrm{minCS}}, \tag{5}$$

where minCS = 1 and maxCS = 10 for a 1–10 survey scale, yielding $\mathrm{NCSS}_m = (10 - \mathrm{score}_m)/9 \in [0, 1]$. The Normalized Customer Satisfaction Score (NCSS) incorporates the often-overlooked factor of customer perception into maintenance prioritization. This component ensures that machines associated with less satisfied customers receive a higher priority, potentially improving overall service quality and customer relationships. This component is properly normalized to [0,1] through its definition and thus excluded from additional min–max normalization.

3.2.3. Operational efficiency metrics

$$\mathrm{BES}_m = \frac{\text{Last labor date}_m - \text{WO open date}_m}{\max_m(\text{Last labor date}_m - \text{WO open date}_m)}. \tag{6}$$

The Booking Efficiency Score (BES) measures the responsiveness of the maintenance process by considering the time between work order creation and service completion. This component incentivizes efficient scheduling and execution of maintenance tasks, with longer booking times resulting in higher priority scores. Like NCSS, BES is already normalized to [0,1] through its definition.

$$\mathrm{APS}_m = \Big[1 + \max\big(0, \min\big(\text{Absolute next target}_m - \mathrm{SMU}_m, \text{Relative next target}_m - \mathrm{SMU}_m\big)\big)\Big]^{-1}, \tag{7}$$

where we clamp negative values (overdue machines) to 0. The APS returns 1 (maximum urgency) if either the absolute or relative target is overdue (i.e., when either target difference becomes negative); otherwise, it decays smoothly with the distance to the nearer target. Finally, the Average Proximity Score (APS) introduces a forward-looking assessment of maintenance urgency. The inverse relationship between APS and the minimum distance to the next maintenance milestone creates an urgency indicator that increases as equipment approaches its next scheduled service. By considering both absolute and relative targets, this component balances adherence to fixed maintenance intervals with equipment-specific usage patterns. Note that APS produces values in (0,1] and is subsequently min–max normalized along with other components for balanced contribution.

Together, these components create a score that captures various operational factors, offering a more nuanced approach to maintenance prioritization. Later in this paper, we build on this priority score by incorporating predictive elements to improve further its effectiveness in prioritizing maintenance tasks.

Table 2
Priority score results for $n = 10$ machines with equal component weighting (standardized values).

Machine	RPI	HPI	NCSS	APS	BES	PS	Rank
m_3	1.00	0.67	0.33	1.00	1.00	0.80	1
m_1	0.00	1.00	0.83	0.49	0.71	0.61	2
m_7	1.00	0.11	0.33	0.46	0.43	0.47	3
m_5	1.00	0.78	0.00	0.00	0.29	0.41	4
m_8	1.00	0.00	0.00	0.70	0.29	0.40	5
m_6	0.00	0.11	0.67	0.89	0.29	0.39	6
m_4	0.00	0.33	1.00	0.07	0.14	0.31	7
m_9	0.00	0.00	0.33	0.93	0.29	0.31	8
m_{10}	0.00	0.22	0.00	0.84	0.14	0.24	9
m_2	0.00	0.00	0.17	0.44	0.00	0.12	10

3.3. Experimental evaluation and sensitivity analysis

We validate our system using a two-year dataset of ten machines, assuming equal weighting ($W_i = 0.2$ for all i) to provide a neutral starting point for evaluation. Table 2 presents the priority scores, ranking Machine 3 highest (0.80), indicating it requires the most urgent attention due to poor performance across multiple factors and illustrating the system's ability to balance multiple factors.

In contrast, traditional pass metrics would have treated machines 1, 4, 6, 9, and 2 identically due to their perfect recent PM accuracy (RPI = 0), despite clear differences in other factors. Our system reveals such subtleties. For instance, machines m_5 and m_8 achieve similar priority scores (0.41 and 0.40) despite having different underlying issues. Both show poor recent PM accuracy (RPI=1.00), but Machine 5's urgency is driven by its poor historical record (HPI=0.78), whereas Machine 8's is driven by its proximity to the next service target interval (APS=0.70), demonstrating the nuanced discrimination our multi-factor approach provides over binary metrics.

To further assess the robustness and flexibility of our priority scoring system, we conduct a sensitivity analysis where we examine four distinct scenarios, each emphasizing different operational priorities: PM impact focus ($W_1 = W_2 = 0.35$), Customer-centric focus ($W_3 = 0.6$), Efficiency focus ($W_5 = 0.6$), and Proximity focus ($W_4 = 0.6$). For each scenario, the remaining weight was equally distributed among other components. To quantify the sensitivity of machine rankings to these weight changes, we introduce a rank volatility metric V_m for each machine m, as shown in the last column of Table 3:

$$V_m = \sum_{s=1}^{4} |r_{m,s} - r_{m,0}|, \tag{8}$$

where $r_{m,s}$ represents the rank of machine m under scenario s, and $r_{m,0}$ is the rank under equal weighting. A higher V_m indicates greater sensitivity to weight changes. Table 3 presents the results of this analysis.

Table 3
Sensitivity analysis results: machine rankings under different weighting scenarios.

Machine	Equal weights	PM impact	Customer-centric	Efficiency	Proximity	V_m
m_3	1	1	5	1	1	4
m_1	2	4	1	2	5	6
m_7	3	2	6	3	3	4
m_5	4	3	10	4	10	13
m_8	5	5	9	5	3	6
m_6	6	10	3	6	2	11
m_4	7	9	2	8	9	10
m_9	8	8	7	7	6	4
m_{10}	9	7	8	8	7	6
m_2	10	6	4	10	8	12

Our analysis revealed several insights. Machines with high priority scores maintained a top priority across most scenarios, reflecting the model's ability to identify critical machines consistently. This capability is particularly valuable for maintenance planning where addressing chronically problematic equipment is important for preventing breakdowns.

m_5 shows the highest volatility ($V_m = 13$), moving from 4th place with equal weights to 3rd under the PM Impact focus, while dropping to 10th under both the Customer-centric and Proximity scenarios. This volatility shows the system's ability to dynamically recalibrate priorities in alignment with shifting operational objectives.

The efficiency focus correctly prioritizes machines with booking delays (e.g., m_3 maintaining 1st place due to its maximum BES score) as the model accounts for process inefficiencies. This scenario could be particularly useful in periods of high demand or resource constraints. The proximity focus considerably reshuffles the rankings, elevating machines like m_6 (from 6th to 2nd) and m_8 (from 5th to 3rd) that are approaching their maintenance deadlines.

The average rank volatility across all machines was 7.6, providing a balance between weight-induced prioritization shifts and scheduling stability. Unlike binary pass/fail metrics, our scoring framework therefore offers three key advantages. It firstly establishes a continuous priority spectrum for finer discrimination between maintenance needs. Second, it permits deliberate component reweighting to match organizational objectives and lastly, it provides quantifiable metrics for resource allocation decisions. This flexibility is particularly valuable in construction environments where equipment criticality and resource constraints fluctuate throughout operational cycles.

With priority scores established, we next develop predictive models to forecast the maintenance variables that inform these priorities.

4. Predictive modeling for maintenance optimization

This section presents our approach to forecasting labor cost, service duration, spare parts cost, and Service Meter Unit (SMU) at repair. We discuss the first three metrics together due to their shared methodological approach and significance in operational planning, while SMU prediction is treated separately due to its distinct role in maintenance scheduling and integration with priority scoring.

4.1. Dataset description

Our dataset, spanning June 2022 to September 2023, includes over 10,000 maintenance events with key attributes such as Service Meter Unit (SMU), service details, costs, estimated durations, and customer information (Table 4).

4.2. Labor cost, service duration and spare parts cost

Accurate forecasts of these variables enhance resource allocation and scheduling (Bevilacqua & Braglia, 2000; Muchiri et al., 2011). We employ a unified machine learning framework to capture interdependencies, consisting of data preprocessing, feature engineering, feature selection, and model training/evaluation (Algorithm 1). By treating these prediction tasks under a unified framework, we aim to capture interdependencies and common patterns that might be overlooked in isolated analyses (Dekker, 1996). We use expanding window (rolling origin) cross-validation to prevent data leakage in our time series predictions.

Algorithm 1 High-Level Prediction Methodology

Require: Time series data $\mathbf{X}$, target variable $\mathbf{y}$, number of cross-validation folds k
Ensure: Optimized model $\mathcal{M}^*$, Predictions $\hat{\mathbf{y}}$
$\mathbf{X}_{\text{clean}}, \mathbf{y}_{\text{clean}} \leftarrow \text{PreprocessData}(\mathbf{X}, \mathbf{y})$
$\mathbf{X}_{\text{features}} \leftarrow \text{EngineerFeatures}(\mathbf{X}_{\text{clean}})$
$\mathbf{X}_{\text{selected}} \leftarrow \text{SelectFeatures}(\mathbf{X}_{\text{features}}, \mathbf{y}_{\text{clean}})$
$\mathbf{X}_{\text{train}}, \mathbf{X}_{\text{test}}, \mathbf{y}_{\text{train}}, \mathbf{y}_{\text{test}} \leftarrow \text{TrainTestSplit}(\mathbf{X}_{\text{selected}}, \mathbf{y}_{\text{clean}})$
$\mathcal{M}^* \leftarrow \text{TrainModels}(\mathbf{X}_{\text{train}}, \mathbf{y}_{\text{train}}, k)$ ▷ Using expanding window CV
$\hat{\mathbf{y}} \leftarrow \text{EvaluateModels}(\mathcal{M}^*, \mathbf{X}_{\text{test}}, \mathbf{y}_{\text{test}})$
return $\mathcal{M}^*, \hat{\mathbf{y}}$

The process begins with data preprocessing to ensure data quality and suitability for time series analysis. Algorithm 2 details this first step.

Algorithm 2 Data Preprocessing

procedure PreprocessData($\mathbf{X}, \mathbf{y}$)
  Remove outliers using IQR method
  Normalize data: $z = (x - \mu)/\sigma$
  Test stationarity (ADF and KPSS tests)
  Apply differencing if non-stationary
  return $\mathbf{X}_{\text{clean}}, \mathbf{y}_{\text{clean}}$
end procedure

This procedure removes outliers using the Interquartile Range (IQR) method (Tukey, 1977), normalizes the data, and ensures stationarity through testing (ADF Dickey & Fuller, 1979 and KPSS Kwiatkowski et al., 1992 tests) and differencing if necessary.

Following preprocessing, we move to feature engineering, an important step in capturing temporal patterns and dependencies in the data. Algorithm 3 outlines this process.

Algorithm 3 Feature Engineering

procedure EngineerFeatures($\mathbf{X}$)
  Create lag features: $x_{t-1}, x_{t-2}, \ldots, x_{t-n}$
  Compute rolling statistics: mean, std, min, max
  Generate date-based features: day of week, month, quarter
  return $\mathbf{X}_{\text{features}}$
end procedure

This procedure generates lag features to capture autoregressive patterns, computes rolling statistics to summarize recent trends, and creates date-based features to account for seasonal and cyclical patterns (Hyndman & Athanasopoulos, 2018). These engineered features provide additional context for the predictive models, potentially improving their accuracy and interpretability.

Table 4
Key attributes of the maintenance dataset.

Attribute	Description
Service call ID	Unique identifier for each service call
Equipment ID	Unique identifier for each piece of equipment
SMU	Service Meter Unit reading at the time of service
SMU last capture date	Date of the most recent SMU reading
Contract type	Type of service contract
Last labor date	Date when the service was performed
Job description	Detailed description of the maintenance task
Component	Specific component serviced or replaced
Category	Classification of service (e.g., PM at 500, 1000, 1500 SMU)
Recorded hours	Actual time taken to complete the service
Estimated hours	Predicted time for service completion
Customer details	Information about the customer (anonymized)
Worker details	Information about the service technician(s)
Labor cost	Actual cost of labor for the service
Estimated labor cost	Predicted cost of labor before service
Spare parts cost	Actual cost of spare parts used
Estimated spare parts cost	Estimated cost of spare parts before service

With an enriched feature set, we then proceed to feature selection to identify the most relevant predictors. Algorithm 4 details this process.

Algorithm 4 Feature Selection

procedure SELECTFEATURES($\mathbf{X}, \mathbf{y}$)
  $\mathbf{F}_{\text{lasso}} \leftarrow \text{LassoRegression}(\mathbf{X}, \mathbf{y}, \alpha)$
  $\mathbf{F}_{\text{rf}} \leftarrow \text{RandomForestImportance}(\mathbf{X}, \mathbf{y})$
  $\mathbf{F}_{\text{step}} \leftarrow \text{StepwiseRegression}(\mathbf{X}, \mathbf{y})$
  $\mathbf{X}_{\text{selected}} \leftarrow \mathbf{F}_{\text{lasso}} \cap \mathbf{F}_{\text{rf}} \cap \mathbf{F}_{\text{step}}$
  return $\mathbf{X}_{\text{selected}}$
end procedure

This procedure combines Lasso regression (Tibshirani, 1996), Random Forest feature importance (Breiman, 2001), and stepwise regression (Hocking, 1976) to select features. The final set of selected features is the intersection of features identified as important by all three methods, ensuring a thorough selection process.

With our data prepared and features selected, we move to the core of our methodology: model training and optimization. Algorithm 5 outlines this process.

This procedure trains four different types of models: ARIMA (Box & Jenkins, 1970), SARIMAX (Brockwell & Davis, 2002), Prophet (Taylor & Letham, 2018), and XGBoost (Chen & Guestrin, 2016). For each model type, a grid search with k-fold cross-validation is performed to find the optimal hyperparameters (Bergstra & Bengio, 2012). The hyperparameter ranges were chosen based on a combination of domain knowledge, literature recommendations (Januschowski et al., 2020), and preliminary experiments. This search ensures each model is well-tuned to the specific characteristics of our maintenance data.

Algorithm 6 Model Evaluation

procedure EVALUATEMODELS($\mathcal{M}^*, \mathbf{X}_{\text{test}}, \mathbf{y}_{\text{test}}$)
  for each optimized model m^* in $\mathcal{M}^*$ **do**
    $\hat{\mathbf{y}}_m \leftarrow \text{Predict}(m^*, \mathbf{X}_{\text{test}})$
    $\text{MAPE}_m \leftarrow \frac{100\%}{n} \sum_{t=1}^{n} \left| \frac{y_t - \hat{y}_t}{y_t} \right|$
    $\text{MSE}_m \leftarrow \frac{1}{n} \sum_{t=1}^{n} (y_t - \hat{y}_t)^2$
    $\text{RMSE}_m \leftarrow \sqrt{\text{MSE}_m}$
    $\text{MAE}_m \leftarrow \frac{1}{n} \sum_{t=1}^{n} |y_t - \hat{y}_t|$
  end for
  $m^* \leftarrow \arg\min_m \text{MAPE}_m$
  return $\hat{\mathbf{y}}_{m^*}$
end procedure

Finally, we evaluate the performance of our optimized models on the test set, as detailed in Algorithm 6. This procedure computes various performance metrics (MAPE, MSE, RMSE, MAE) for each optimized model on the test set (Hyndman & Koehler, 2006), as detailed in Algorithm 6.

Algorithm 5 Model Training and Optimization

procedure TRAINMODELS($\mathbf{X}_{\text{train}}, \mathbf{y}_{\text{train}}, k$)
  $\Theta_{\text{ARIMA}} \leftarrow \{$
    $p \in \{0, 1, 2, 4, 6\}$,
    $d \in \{0, 1, 2\}$,
    $q \in \{0, 1, 2, 4\}$
  $\}$
  $\Theta_{\text{SARIMAX}} \leftarrow \{$
    $p \in \{0, 1, 2\}, d \in \{0, 1\}, q \in \{0, 1, 2\}$,
    $P \in \{0, 1\}, D \in \{0, 1\}, Q \in \{0, 1\}$,
    $m \in \{12, 52\}$ ▷ For monthly and weekly seasonality
  $\}$
  $\Theta_{\text{Prophet}} \leftarrow \{$
    changepoint_prior_scale $\in \{0.001, 0.01, 0.1, 0.5\}$,
    seasonality_prior_scale $\in \{0.01, 0.1, 1.0, 10.0\}$
  $\}$
  $\Theta_{\text{XGBoost}} \leftarrow \{$
    max_depth $\in \{3, 6, 10\}$,
    learning_rate $\in \{0.01, 0.1, 0.3\}$,
    n_estimators $\in \{100, 500, 1000\}$,
    min_child_weight $\in \{1, 3, 5\}$
  $\}$
  for each model type $m \in$ {ARIMA, SARIMAX, Prophet, XGBoost} **do**
    $\mathcal{M}_m^* \leftarrow \text{GridSearchCV}(m, \Theta_m, \mathbf{X}_{\text{train}}, \mathbf{y}_{\text{train}}, k)$
  end for
  return $\{\mathcal{M}^*_{\text{ARIMA}}, \mathcal{M}^*_{\text{SARIMAX}}, \mathcal{M}^*_{\text{Prophet}}, \mathcal{M}^*_{\text{XGBoost}}\}$
end procedure

4.2.1. Model performance and comparative analysis

We now turn our attention to the performance of our models in forecasting labor costs, service durations, and spare parts costs, and compare these results with the company's existing estimation practices.

The preprocessing stage, as detailed in Algorithm 2, was important in preparing our data for analysis. For labor cost, outlier removal using the IQR method reduced the maximum value from €60,051.25 to €1,272.39. Service duration data initially deviated from normality but achieved a normal distribution after outlier removal (Shapiro–Wilk test, p-value: 0.1514), while spare parts cost required log transformation to address its right-skewed distribution.

Time series decomposition and feature engineering, following Algorithm 3, showed distinct patterns for each prediction target. Labor cost showed a downward trend with regular monthly fluctuations, service duration fluctuated between 250-500 min with a notable decline near

Table 5
Performance metrics for prediction models across all tasks.

Target variable	Model	MSE	MAE	RMSE	Train MAPE (%)	Test MAPE (%)
Labor Cost	SARIMAX	2,849,401.33	1,329.40	1,687.40	2.73	4.24
	Prophet	25,477,421.97	4,498.90	5,047.53	13.26	15.82
	ARIMA	34,951,850.46	4,288.39	5,911.17	12.14	13.99
	Baseline	52,555,250.85	6,583.39	7,249.42	–	22.01
Service Duration	SARIMAX	5,268.04	56.75	68.93	7.48	9.32
	Prophet	5,537.80	57.42	73.28	15.71	18.04
	XGBoost	7,007.89	58.81	78.68	26.92	30.79
	Baseline	16,302.74	112.04	126.79	–	37.76
Spare Parts Cost	SARIMAX	15,624.32	98.56	125.00	10.68	12.45
	Prophet	18,769.01	107.23	137.00	12.91	14.87
	XGBoost	20,164.49	110.89	142.00	13.27	16.32
	Baseline	42,436.81	162.34	206.00	–	28.76

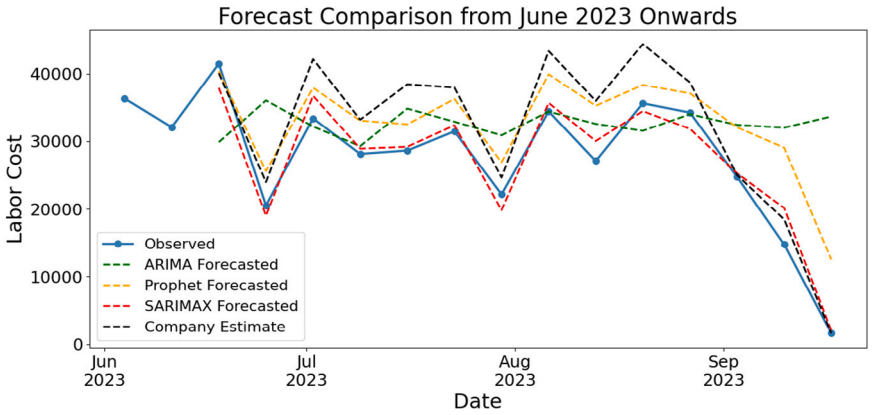

Fig. 2. Comparison of observed labor costs with forecasts from ARIMA, Prophet, SARIMAX models, and company estimates from June 2023 onwards.

the observation period's end, and spare parts cost displayed cyclical patterns aligning with maintenance intervals.

Feature selection, implemented as per Algorithm 4, identified key predictors for each task. For labor cost, estimated and actual spare parts costs were most significant. Service duration was strongly influenced by the number of workers and estimated spare parts cost. Spare parts cost prediction relied heavily on equipment age, usage intensity, and historical maintenance records.

Table 5 presents the averaged results of the performance metrics for ARIMA, SARIMAX, Prophet, and XGBoost models across all five folds for the three prediction tasks, compared against the company's baseline estimates.

For labor cost prediction, the SARIMAX model achieved the lowest error rates across all metrics, with a MAPE of 4.24%. Prophet and ARIMA models also improved upon the baseline estimates, with MAPEs of 15.82% and 13.99% respectively, compared to the baseline MAPE of 22.01%. The train-test MAPE gaps ranging from 1.51 to 3.87 percentage points across all models suggest healthy generalization, as they are large enough to show the models are not memorizing training data, yet small enough to confirm they capture genuine patterns effectively. These results were validated using a Wilcoxon signed-rank test on absolute errors ($p < 0.01$), confirming statistically significant improvements over baseline methods.

Fig. 2 compares model forecasts against actual labor cost values from June 2023 onward. SARIMAX closely aligns with observed data, confirming its superior accuracy (Table 5), while Prophet captures general trends but misses key fluctuations, particularly the decline seen in September 2023. ARIMA shows less responsiveness, especially from late August onward. Company estimates, while consistently overestimating costs, capture some overall patterns, indicating room for refinement but lacking precision.

For service duration forecasting, SARIMAX achieved 9.32% MAPE, representing a 75.32% error reduction compared to baseline methods (37.76%). Similarly, SARIMAX reduced spare parts cost prediction error to 12.45% MAPE, a 56.71% improvement over the baseline (28.76%).

Diagnostic validation confirmed SARIMAX model robustness. Labor cost predictions showed standardized residuals within the [−3,3] interval, with normality verified through Q-Q plots, Jarque–Bera test (Jarque & Bera, 1980) (p-value: 0.93), and Ljung–Box test (Ljung & Box, 1978) (p-value: 0.75). Service duration residuals generally ranged from −2 to 3, with minor Q-Q plot tail deviations suggesting potential areas for model refinement. Spare parts cost residuals showed a similar pattern, with slight heteroscedasticity at higher predicted values.

While SARIMAX outperforms traditional estimates, incorporating additional features or advanced models (e.g., LSTM, TCN) could further capture non-linear dependencies and complex temporal patterns. Next, we focus on predicting SMU at repair, leveraging historical maintenance patterns to improve scheduling accuracy.

4.3. Predicting the next likely maintenance event

As discussed in Section 3, the heavy construction equipment industry typically schedules maintenance at 500 SMU intervals. However, maintaining these intervals within the targeted ±50 SMU range is challenging due to operational factors such as variable worksite conditions, operator behaviors, fluctuating project schedules, and equipment availability constraints, which can delay maintenance and impact reliability.

We propose a machine learning approach using historical maintenance data to improve maintenance timing predictions. This method captures individual excavator maintenance patterns without requiring additional machine health data, relying solely on past service records, yet still showing high accuracy.

Since SMU values naturally increase over time, a simple linear regression model for each machine can predict the next maintenance

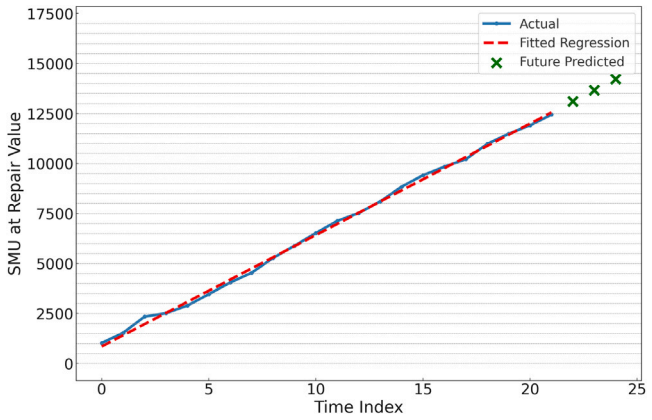

Fig. 3. Example of a linear regression model fitted to a single machine's historical maintenance data.

SMU based on historical trends. Fig. 3 illustrates such a model applied to a single machine.

The next three predicted values, shown as green crosses, are 13,105, 13,662, and 14,219. To meet pass metrics targets, these SMU values should remain within ±50 SMU of 500 intervals. However, the forecasts from Fig. 3 result in AP scores of 0, 0, and 0, respectively.

While the regression model reflects historical maintenance performance, it may not be useful for planning, as service companies need to know when and at which SMU maintenance should occur to meet targets and minimize downtime. Additionally, service targets based on AP, RP, and BCP metrics impact yearly performance and budgets. This creates a trade-off where models accurately reflecting historical patterns may perpetuate suboptimal maintenance timing rather than guiding improvement. Optimizing AP increases interval adherence but can worsen RP (or vice-versa); we target a balanced improvement measured by BCP.

To address this, we implement a custom loss function that penalizes predictions deviating from the 500 SMU target, with higher penalties for deviations beyond ±50 SMU. This aligns predictions with desired maintenance intervals, improving target achievement. The custom loss function is defined in Algorithm 4.3.

Algorithm 7 Custom Loss Function for SMU Prediction

Input: True SMU values $\mathbf{y}$, Predicted SMU values $\hat{\mathbf{y}}$
Output: Custom loss value
for each i in $\{1, 2, \ldots, N\}$ **do**
  Compute the next nearest multiple of 500 for $\hat{y}_i$: $nearest\_500_i = 500 \times \text{round}(\hat{y}_i/500)$
  Compute the deviation: $deviation_i = |\hat{y}_i - nearest\_500_i|$
  if $deviation_i > 50$ **then**
    Compute the penalty: $penalty_i = (deviation_i - 50)^3$
  else
    $penalty_i = 0$
  end if
end for
Compute the custom loss: $Loss = \frac{1}{N} \sum_{i=1}^{N} penalty_i$
Return: $Loss$

The function calculates the deviation of the predicted SMU from the nearest multiple of 500 using round-to-nearest (0.5 up) convention. No penalty is applied within ±50 SMU, while a cubic penalty is imposed beyond this range. This ensures small deviations receive modest penalties while larger discrepancies face exponentially increasing penalties, reflecting the rising risks and costs of poorly timed maintenance. The final loss is computed as the mean of these penalties across all data points, guiding model training to better align with the desired maintenance schedule.

Table 6
Performance comparison of SMU prediction models.

Model	RMSE	MAE	AP Success	RP Success	BCP success
Observed	–	–	40%	40%	50%
ARIMA	152.6	118.3	20%	90%	90%
Average Slope	147.2	112.5	30%	80%	80%
Linear Regression	165.8	129.7	20%	90%	90%
LightGBM	89.4	67.2	60%	70%	90%
LSTM	93.7	71.5	60%	40%	70%

We integrate the custom loss function into LightGBM and LSTM architectures, due to their ability to capture complex temporal patterns while accommodating custom training objectives.

For comparison, we include three additional models: ARIMA, Average Slope Projection, and Linear Regression with time features. ARIMA handles time series data, Average Slope Projection serves as a baseline using average SMU change, and Linear Regression incorporates temporal features like day of the week and elapsed time. We use an 80/20 train-test split, with results in Fig. 4 from the test set.

The left plot in Fig. 4 shows observed vs. predicted SMU values, though overlapping predictions make model comparison difficult. The right plot highlights deviations over time, revealing an underestimation in August 2023 across all models, likely due to unexpected machine usage or maintenance patterns. ARIMA and Average Slope Projection produce similar results, initially accurate but overestimating SMU by late 2023.

The Linear Regression model initially deviates by up to −300 SMU, struggling with early temporal patterns but improving over time. LightGBM and LSTM, incorporating the custom loss function, perform similarly, with small initial deviations (∼50 SMU) and a dip in August 2023, but with improved accuracy by early 2024.

While these forecasts show prediction accuracy, they do not indicate when maintenance should occur to meet pass metrics, reinforcing the need for the custom loss function. Table 6 compares models based on traditional error metrics (RMSE, MAE) and maintenance-specific success rates (AP, RP, BCP).

The LightGBM and LSTM models, incorporating the custom loss function, achieve the lowest RMSE and MAE values while attaining the highest AP success rate (60%), compared to 40% in the observed data. This suggests the custom loss function effectively aligns predictions with the ideal 500 SMU intervals. In contrast, ARIMA and Linear Regression show lower AP performance (20%), indicating difficulty in meeting absolute interval targets.

The RP metric, which measures maintenance within the ±50 SMU tolerance, highlights different model strengths. ARIMA and Linear Regression achieve 90% RP success (vs. 40% observed), benefiting from stable SMU interval changes despite lower absolute accuracy. The Average Slope method follows with 80%, LightGBM with 70%, and LSTM with 40%, despite its strong AP performance.

For the BCP metric, which considers either AP or RP success, ARIMA, Linear Regression, and LightGBM achieve 90% success (vs. 50% observed). The Average Slope method follows with 80%, while LSTM reaches 70%, indicating that different error profiles can still lead to similar scheduling success.

These findings give insight into trade-offs between prediction accuracy and maintenance scheduling. LightGBM balances low error metrics with high success rates, making it a strong overall choice. However, if precise SMU predictions are the priority, LightGBM or LSTM may be preferred. At the same time, ARIMA or Linear Regression may be better suited for maintaining consistent relative intervals despite higher RMSE and MAE values.

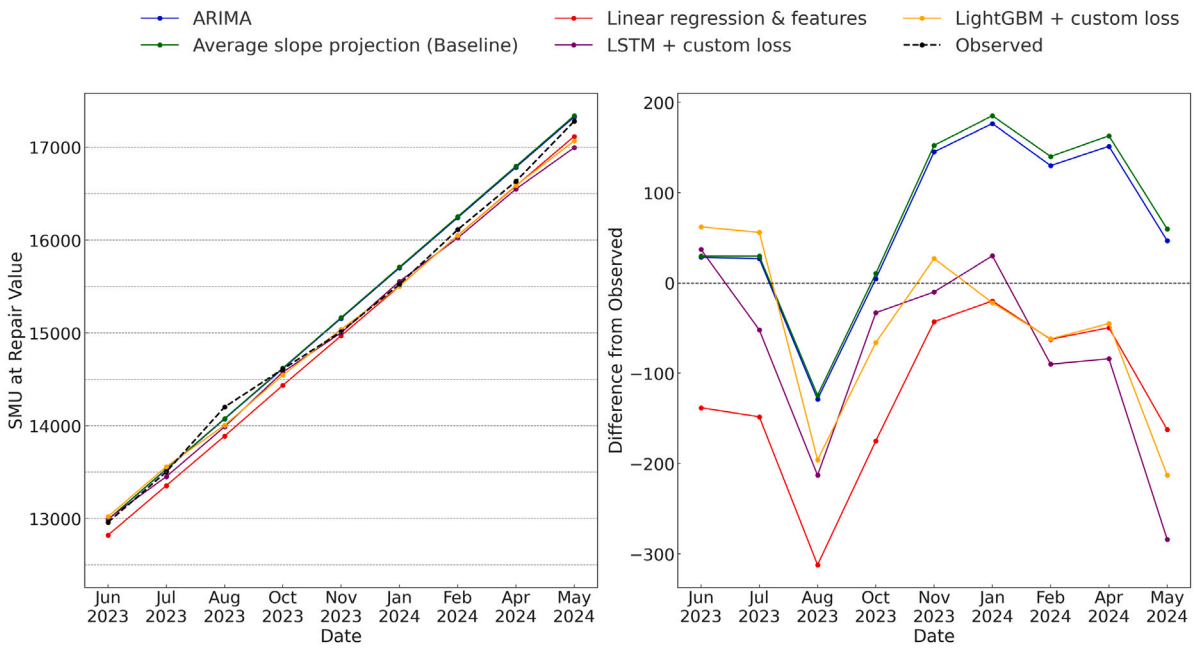

Fig. 4. The left plot shows the observed versus predicted SMU values for different models for a single machine. The right plot shows the difference of the observed and predicted for different models over time.

4.3.1. Incorporation of SMU predictions into priority score

Building on the SMU prediction models from the previous section, we now integrate these forecasts into the priority scoring system established in Section 3.2. This integration creates a forward-looking dimension that enhances scoring accuracy by considering both historical performance patterns and projected maintenance needs.

We formalize this approach by introducing the predicted maintenance urgency (PMU) metric, which quantifies the temporal proximity to the next predicted maintenance event. Let SMU_{pred} be the predicted next maintenance SMU and SMU_{curr} the current SMU. The PMU is computed using an exponential decay function that peaks at 1 when the machine is at its predicted maintenance point and decreases smoothly as the machine moves away from that point:

$$\text{PMU} = \exp\left(-\frac{|\text{SMU}_{\text{pred}} - \text{SMU}_{\text{curr}}|}{c}\right), \tag{9}$$

where c controls the decay rate. We set $c = 500$ to align with the standard 500-SMU maintenance interval. This function ensures PMU = 1 at the predicted maintenance point ($\text{SMU}_{\text{curr}} = \text{SMU}_{\text{pred}}$) and decays symmetrically for machines that are either approaching or have passed their maintenance point.

It is important to note that APS measures closeness to the next fixed manufacturer-specified interval, while PMU reflects proximity to the model-predicted maintenance event. We retain both components to balance adherence to manufacturer intervals with data-inferred operating cadence. In our experiments, we set both weights equally ($W_4 = W_6 = 0.167$) with the remaining weights distributed uniformly among other components, though these can be adjusted based on organizational priorities.

Building on Eq. (2), we incorporate PMU into the updated priority score:

$$\begin{aligned}\text{PS}_m = &W_1 \cdot \tilde{\text{RPI}}_m + W_2 \cdot \tilde{\text{HPI}}_m + \\ &W_3 \cdot \text{NCSS}_m + W_4 \cdot \tilde{\text{APS}}_m + \\ &W_5 \cdot \text{BES}_m + W_6 \cdot \tilde{\text{PMU}}_m,\end{aligned} \tag{10}$$

where W_6 represents the weight assigned to the PMU component. The choice of W_6 should reflect the importance of the predicted maintenance urgency relative to the other components where the $\sum W_n = 1$.

Table 7
Comparative analysis of priority scores across ten machines.

Machine	SMU_{curr}	SMU_{pred}	$\text{SMU}_{\text{actual}}$	PS_{orig}	PS_{pred}
1	8,523	8,800	9,015	0.35	0.42
2	12,342	12,620	12,754	0.27	0.33
3	6,789	7,075	7,098	0.42	0.47
4	10,054	10,250	10,503	0.30	0.36
5	9,204	9,970	9,712	0.33	0.40
6	11,598	12,020	12,043	0.29	0.34
7	8,802	9,160	9,306	0.36	0.41
8	7,612	8,230	8,109	0.38	0.43
9	10,245	10,370	10,689	0.31	0.38
10	11,010	11,730	11,543	0.32	0.37

To assess the impact of SMU predictions, we compare the original (PS_{orig}) and updated (PS_{pred}) priority scores using the dataset from Section 3.3. We introduce the Predictive Maintenance Improvement (PMI) metric to quantify predictive accuracy:

$$\text{PMI} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{|\text{SMU}_{\text{actual},i} - \text{SMU}_{\text{pred}}|}{|\text{SMU}_{\text{actual},i} - \text{SMU}_{\text{curr}}|} - 1\right), \tag{11}$$

where $\text{SMU}_{\text{actual},i}$ is the actual maintenance SMU for machine i. Negative PMI values indicate improvement over the naive baseline, with the magnitude representing percentage reduction in prediction error. Table 7 presents the results of the comparative analysis for a sample of ten machines.

The PMI value calculated from the data in Table 7 is −0.644, indicating a 64.4% improvement in predictive capability for these ten evaluation machines, suggesting that the updated priority scores (PS_{pred}) are more effective in anticipating future maintenance needs compared to the original scores (PS_{orig}).

To further validate the statistical significance of the improvement, we perform a paired t-test on the absolute deviations between the predicted and actual maintenance events for both the original and updated priority scores. The null hypothesis (H_0) states that there is no significant difference in the mean absolute deviations, while the alternative hypothesis (H_1) states that the mean absolute deviation is significantly lower for the updated priority scores.

$$H_0 : \mu_{\text{orig}} - \mu_{\text{pred}} = 0,$$

$H_1 : \mu_{\text{orig}} - \mu_{\text{pred}} > 0.$

The paired t-test resulted in a t-statistic of 10.41 and a one-sided p-value of approximately 1.28×10^{-6}, indicating strong evidence to reject the null hypothesis in favor of the alternative. This result confirms that the incorporation of SMU predictions into the priority scoring system leads to a statistically significant improvement in predicting future maintenance needs.

Having developed accurate predictions for maintenance requirements, we now address the challenge of optimally routing technicians to service locations.

5. Routing model

While accurate prediction of maintenance requirements including labor costs, service duration, spare parts costs, and service urgency forms the foundation of optimization, efficient workforce deployment remains a critical challenge for service companies managing geographically dispersed equipment fleets. Strategic routing of maintenance technicians reduces travel time, minimizes equipment downtime, improves response rates, and ensures compliance with service level agreements.

This section introduces a routing optimization framework that leverages optimization and heuristic approaches, including a neural network method not previously applied to maintenance routing problems. The model incorporates the predictive insights developed in earlier sections to enhance personnel allocation and scheduling decisions, thereby reducing operational costs while ensuring timely service delivery.

5.1. Problem definition

The maintenance routing problem involves efficiently scheduling technicians who begin and end their workdays at home locations while servicing multiple geographically dispersed equipment sites. This operational challenge requires minimizing total time (both travel and service) while respecting shift duration constraints. We therefore model this as a Multi-Depot Capacitated Vehicle Routing Problem (MDCVRP), where technicians' homes function as depots, service locations represent customers, and shift durations constitute capacity constraints.

Fig. 5 illustrates an example MDCVRP solution for heavy equipment maintenance. Hexagons (D_i) represent mechanics' homes (depots), squares (M_j) denote machine locations (customers), and colored routes indicate different mechanics' paths. Service times (s_j) and travel times (d_{ij}) are displayed along routes, reflecting maintenance duration and travel constraints.

To illustrate a solution, consider Mechanic 1's route (blue). Starting at depot D_1, the mechanic travels to M_1 ($d_{D1,1} = 49$ min), performs maintenance for $s_1 = 81$ min, then proceeds to M_2 ($d_{1,2} = 67$ min, $s_2 = 44$ min) and M_3 ($d_{2,3} = 45$ min, $s_3 = 101$ min), before returning to D_1 ($d_{3,D1} = 33$ min). The total route time (T_1) is 420 min, with 194 min of travel and 226 min of service. Similarly, Mechanic 2 (orange) visits M_4 and M_5, completing a 354-minute route (T_2). Mechanic 3 (green) services M_6, M_7, and M_8, totaling 404 min (T_3).

The MDCVRP minimizes total route time ($T_1 + T_2 + T_3$), including travel and service durations, while adhering to shift limits. In this example, all routes satisfy this constraint, though Mechanic 3's route approaches the limit. The next subsection formally defines the problem and its constraints.

5.2. Mathematical formulation

We formulate the maintenance routing problem as a mixed-integer linear programming (MILP) model with priority considerations. Let $N = \{1, 2, \dots, n\}$ be the set of customer locations, and $D = \{n+1, \dots, n+m\}$ be the set of depot locations. Let $V = \{1, 2, \dots, v\}$ be the set of vehicles (mechanics). Define $A = \{(i, j) : i, j \in N \cup D, i \neq j\}$ as the set of all possible arcs. The binary decision variable x_{ijk} equals 1 if vehicle k travels from node i to node j in its assigned shift, and 0 otherwise. The binary decision variable y_k equals 1 if vehicle k is used, and 0 otherwise. The continuous decision variable u_{ik} represents the position of customer i in the route of vehicle k (per-vehicle MTZ formulation). The parameters of the model are as follows:

- $d_{ij} \geq 0$: travel time from node i to node j, $\forall (i, j) \in A$
- $s_i \geq 0$: service time at node i, $\forall i \in N$ (with $s_j = 0$ for depots $j \in D$)
- $T_{ij} = d_{ij} + s_j$: total time (travel plus service at destination) from node i to node j, $\forall (i, j) \in A$
- $M > 0$: maximum working hours for each vehicle (shift duration)
- $depot(k) \in D$: the assigned depot for vehicle k, $\forall k \in V$
- $p_i \in [0, 1]$: normalized priority score of customer i for $i \in N$, with $p_j = 0$ for depots $j \in D$
- $\lambda \in [0, 1]$: weighting factor for priority scores (dimensionless, kept modest to ensure $(T_{ij} - \lambda p_j) > 0$)
- $P_{min} > 0$: minimum sum of priority scores per route

The MDCVRP can be formulated as follows:

$$\min \quad \sum_{(i,j)\in A} \sum_{k\in V} (T_{ij} - \lambda p_j) x_{ijk} \tag{12}$$

$$\text{s.t.} \quad \sum_{j\in N\cup D} \sum_{k\in V} x_{ijk} = 1, \quad \forall i \in N \tag{13}$$

$$\sum_{j\in N\cup D} x_{ojk} = \sum_{i\in N\cup D} x_{iok} = y_k, \quad \forall k \in V,\ o = depot(k) \tag{14}$$

$$\sum_{i\in N\cup D} x_{ijk} - \sum_{i\in N\cup D} x_{jik} = 0, \quad \forall j \in N,\ k \in V \tag{15}$$

$$u_{ik} - u_{jk} + |N|\, x_{ijk} \leq |N| - 1, \quad \forall i, j \in N,\ i \neq j,\ k \in V \tag{16}$$

$$u_{ik} \leq |N| \sum_{j\in N\cup D} x_{ijk}, \quad \forall i \in N,\ k \in V \tag{17}$$

$$u_{ik} \geq \sum_{j\in N\cup D} x_{ijk}, \quad \forall i \in N,\ k \in V \tag{18}$$

$$\sum_{(i,j)\in A} T_{ij} x_{ijk} \leq M y_k, \quad \forall k \in V \tag{19}$$

$$x_{ijk} \leq y_k, \quad \forall (i, j) \in A,\ k \in V \tag{20}$$

$$x_{ijk} = 0, \quad \forall i, j \in D,\ i \neq j,\ k \in V \tag{21}$$

$$\sum_{j\in N} x_{djk} = 0, \quad \forall k \in V,\ \forall d \in D \setminus \{depot(k)\} \tag{22}$$

$$\sum_{i\in N} x_{idk} = 0, \quad \forall k \in V,\ \forall d \in D \setminus \{depot(k)\} \tag{23}$$

$$\sum_{j\in N\cup D} x_{ijk} \leq 1, \quad \forall i \in N \cup D,\ k \in V \tag{24}$$

$$\sum_{i\in N\cup D} x_{ijk} \leq 1, \quad \forall j \in N \cup D,\ k \in V \tag{25}$$

$$\sum_{i\in N} p_i \sum_{j\in N\cup D} x_{ijk} \geq P_{min} y_k, \quad \forall k \in V \tag{26}$$

$$x_{ijk} \in \{0, 1\}, \quad \forall (i, j) \in A,\ k \in V \tag{27}$$

$$y_k \in \{0, 1\}, \quad \forall k \in V \tag{28}$$

$$0 \leq u_{ik} \leq |N|, \quad \forall i \in N,\ k \in V \tag{29}$$

The objective function (12) minimizes the total route time while incorporating machine priority scores through a weighted approach. The term T_{ij} represents the combined travel and service time between nodes (with service time counted at the destination), while λp_j adjusts for priority, effectively reducing the perceived cost of visiting high-priority machine j (not leaving machine i). Since $p_j \in [0, 1]$ after normalization (with $p_j = 0$ for depots), the parameter $\lambda \in [0, 1]$ controls the relative importance of priority versus time minimization while ensuring $(T_{ij} - \lambda p_j) > 0$ for all arcs. In our main experiments, we use $\lambda = 0.4$ based on the sensitivity analysis presented in Section 5.3.2.

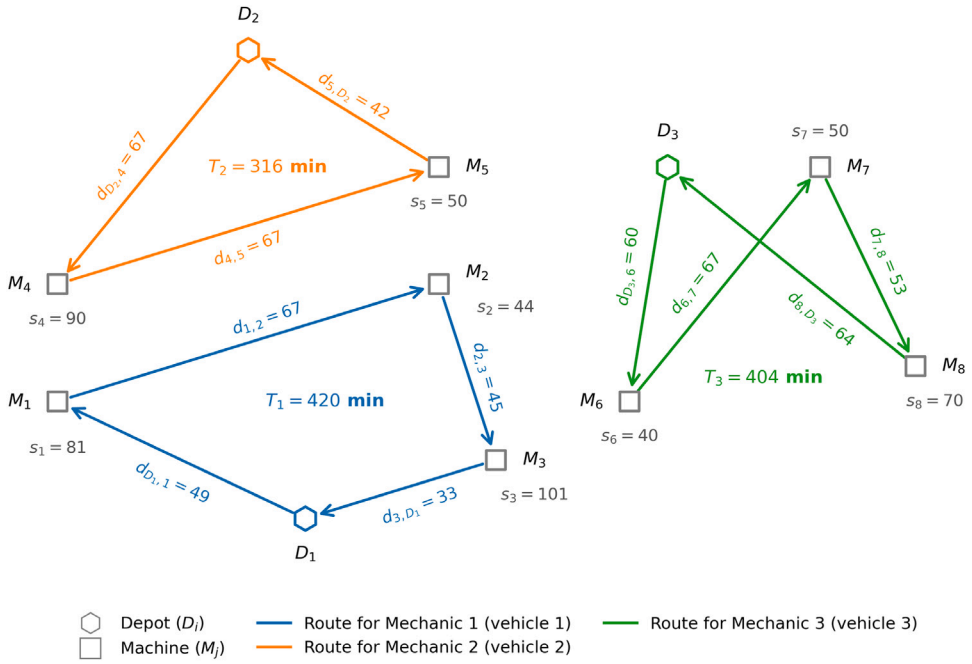

Fig. 5. Illustration of an MDCVRP for heavy equipment maintenance. The figure shows three depots (D_i), each servicing nearby machines (M_j) with service times (s_j) and travel times (d_{ij}). Example solution for three mechanics are displayed, each with their total route time (T_k).

Constraint set (13) ensures each customer is visited exactly once. Constraint set (14) enforces that each vehicle departs from and returns to its assigned depot $depot(k)$. Constraint set (15) maintains route continuity by ensuring each vehicle exiting a customer node also re-enters it. To prevent infeasible routes, constraint set (16) applies the Miller–Tucker–Zemlin (MTZ) formulation with per-vehicle position variables u_{ik}, ensuring proper sequencing within each vehicle's route. Constraint set (17) links these position variables to the routing decisions, and (18) strengthens the link by forcing u_{ik} to be strictly positive whenever customer i is served by vehicle k (and zero otherwise). Constraint set (19) limits total route time (T_{ij}) to the predefined shift duration M, aligning with the T_k values in the problem definition. Constraint sets (22)–(23) disallow visits to depots other than a vehicle's assigned depot, preventing detours through non-assigned depots.

Additional constraints enforce feasibility: (20) links arc usage (x_{ijk}) with vehicle usage (y_k), ensuring only active vehicles traverse arcs; (21) prevents direct travel between depots; (24) and (25) enforce that each node has at most one outgoing and one incoming arc per vehicle. To ensure priority balance, constraint set (26) requires that each vehicle's served customers meet a minimum sum of priority scores P_{min}, which enforces a minimum total priority workload per route rather than an average. Finally, (27) and (28) define the binary nature of decision variables x_{ijk} and y_k, while (29) sets bounds for auxiliary variables u_{ik} in subtour elimination.

5.3. Solution approach

Solving the MDCVRP to optimality using exact methods is computationally challenging due to its NP-hard nature (Milan et al., 2017). The problem's complexity grows exponentially with instance size, making exact methods impractical for real-world maintenance scheduling scenarios. To address this limitation, we propose a heuristic-based solution framework that efficiently generates high-quality routes while incorporating priority information from our predictive models.

Our approach implements a modified cluster-first, route-second (CFRS) strategy that decomposes the original problem into manageable subproblems. The key innovation lies in integrating priority scores throughout both construction and refinement phases, ensuring critical maintenance tasks receive timely service. These priority scores, derived from Eq. (10), directly incorporate both historical maintenance performance metrics and forward-looking SMU predictions, enabling maintenance scheduling that aligns with both immediate needs and future equipment conditions. Algorithm 8 presents our two-stage solution methodology.

Algorithm 8 Modified CFRS approach

Require: Set of mechanics $\mathcal{M}$, mechanic capacity Q, set of machines $\mathcal{N}$, predicted priority scores $\mathcal{P} = \{p_i \mid i \in \mathcal{N}\}$, number of clusters K
Ensure: Set of routes $\mathcal{R}$ for all clusters
// **Construction Stage**
$\mathcal{C} \leftarrow$ KMeansClustering($\mathcal{N}, K$)
for each cluster $C_j \in \mathcal{C}$ **do**
  $R_j \leftarrow$ PriorityGreedyConstruction($C_j, \mathcal{M}, Q, \mathcal{P}$)
end for
// **Routing Stage**
$\mathcal{E} \leftarrow$ SelectExploitationTechnique(ALNS, 2-OPT, SOM)
for each cluster $C_j \in \mathcal{C}$ **do**
  $R_j \leftarrow$ ApplyExploitationTechnique($R_j, \mathcal{E}, \mathcal{M}, Q, \mathcal{P}$)
end for
$\mathcal{R} \leftarrow$ CombineBestRoutes($\mathcal{C}, \{R_j \mid C_j \in \mathcal{C}\}$)
return $\mathcal{R}$

In the construction stage, the algorithm partitions machines $\mathcal{N}$ into K clusters using KMeansClustering. For each cluster C_j, PriorityGreedyConstruction generates initial routes R_j based on predicted priority scores $\mathcal{P}$ and mechanic capacity constraints, ensuring high-priority machines are served first while balancing workload distribution.

In the routing stage, the initial solution is refined using one of three exploitation techniques: Adaptive Large Neighborhood Search (ALNS), 2-OPT, or a modified Self-Organizing Maps (SOM) algorithm. The user selects a technique via SelectExploitationTechnique, which is applied to each cluster's routes R_j using ApplyExploitationTechnique, considering mechanics $\mathcal{M}$, capacity Q, and priority scores $\mathcal{P}$. These techniques explore neighborhood structures and local search moves to improve routing while maintaining priority constraints, as detailed in Section 5.3.2.

Finally, CombineBestRoutes merges the best-found routes from each cluster, producing the final MDCVRP solution $\mathcal{R}$. The algorithm returns this optimized set of routes as the final output.

5.3.1. Construction stage

In the initial solution construction stage, we use a hybrid approach combining k-means clustering and a priority-based greedy insertion strategy. K-means clustering partitions machines into geographically compact clusters, reducing problem complexity and optimizing routes by minimizing inter-cluster travel times and distances.

Within each cluster, the priority-based greedy insertion strategy constructs initial routes, prioritizing high-priority machines to reduce downtime and maintain productivity. This method considers predicted priority scores and mechanic capacity constraints to generate feasible, optimized routes. The pseudocode for this stage is presented in Algorithm 9.

Algorithm 9 Construction Stage

Require: Set of mechanics $\mathcal{M}$, mechanic capacity Q, set of customer locations $\mathcal{N}$, predicted priority scores $\mathcal{P} = \{p_i \mid i \in \mathcal{N}\}$, number of clusters K
Ensure: Set of routes $\mathcal{R}$ for all clusters
$\mathcal{C} \leftarrow \emptyset$
Initialize empty clusters: $C_1, C_2, \ldots, C_K$
$\mathcal{C} \leftarrow \{C_1, C_2, \ldots, C_K\}$
$centroids \leftarrow$ InitializeCentroids(K)
while not converged **do**
  for each location $i \in \mathcal{N}$ **do**
    $c_i \leftarrow \arg\min_{c \in centroids} d(i, c)$
    Assign location i to cluster C_{c_i}
  end for
  for each cluster $C_j \in \mathcal{C}$ **do**
    $centroids[j] \leftarrow \frac{1}{|C_j|} \sum_{i \in C_j} \text{Location}(i)$
  end for
end while
for each cluster $C_j \in \mathcal{C}$ **do**
  Sort locations in C_j by descending priority score p_i
  $R_j \leftarrow \emptyset$
  for each location $i \in C_j$ **do**
    $k^* \leftarrow \arg\min_{k \in \mathcal{M}} T_{\text{depot}(k),i}$
    if $\sum_{(l,m) \in R_{jk^*}} T_{lm} + T_{\text{depot}(k^*),i} \leq Q$ **then**
      Append location i to route R_{jk^*}
    else
      Create new route for location i
    end if
  end for
  $\mathcal{R} \leftarrow \mathcal{R} \cup \{R_j\}$
end for
return $\mathcal{R}$

The construction stage takes as input the set of mechanics $\mathcal{M}$, mechanic capacity Q, machines $\mathcal{N}$, predicted priority scores $\mathcal{P}$, and the desired number of clusters K. It initializes an empty set of clusters $\mathcal{C}$ and randomly selects K centroids. The k-means clustering algorithm iteratively assigns machines to the nearest centroid based on distance $d(i, c)$ and updates centroids until convergence.

Once clusters are formed, priority greedy construction is applied within each cluster C_j. Machines are sorted in descending order of priority scores p_i, directly incorporating the predictive maintenance urgency metrics developed in Section 4.3.1. This sorting ensures that machines with higher PMU values and overall priority scores receive scheduling preference regardless of their geographic position. For each machine i, the nearest mechanic k^* is selected to minimize travel plus service time. If adding i to mechanic k^*'s route (R_{jk^*}) does not exceed capacity Q, it is assigned; otherwise, a new route is created. This process continues until all machines are assigned.

The final set of routes $\mathcal{R}$, combining routes R_j from all clusters, is returned as the initial solution.

5.3.2. Routing stage

The routing stage refines the initial solution by exploring better routing options within each cluster while preserving task priority. We apply three intensification strategies—Adaptive Large Neighborhood Search (ALNS), 2-OPT, and a modified Self-Organizing Maps (SOM) algorithm. Each strategy targets specific aspects of solution improvement and is applied independently to the initial cluster routes.

- **Adaptive Large Neighborhood Search (ALNS):** ALNS improves the initial routes within each cluster by iteratively destroying and repairing the solution. Algorithm 10 presents the pseudocode for ALNS.

Algorithm 10 Adaptive Large Neighborhood Search (ALNS)

Require: Initial routes R_j for cluster C_j, set of mechanics $\mathcal{M}$, mechanic capacity Q, predicted priority scores $\mathcal{P} = \{p_i \mid i \in C_j\}$, iteration limit I_{max}
Ensure: Improved routes R_j^* for cluster C_j
$R_j^* \leftarrow R_j$, $i \leftarrow 0$
while $i < I_{max}$ **do**
  $R_j' \leftarrow$ DestroyRepair($R_j^*, \mathcal{M}, Q, \mathcal{P}$)
  if $f(R_j') < f(R_j^*)$ **then** ▷ $f(R)$ calculates total time of routes
    $R_j^* \leftarrow R_j'$
  end if
  UpdateWeights(), $i \leftarrow i + 1$
end while
return R_j^*

ALNS takes the initial routes R_j, the set of mechanics $\mathcal{M}$, mechanic capacity Q, predicted priority scores $\mathcal{P}$, and the iteration limit I_{max} as input. The DestroyRepair function removes a subset of service locations from the routes and reinserts them in a different order or assigns them to different mechanics, guided by adaptive weights. Soft constraints are incorporated by penalizing changes affecting the order of high-priority service locations. If the new solution R_j' has a better objective function value $f(R_j')$ than the current best solution R_j^*, R_j^* is updated with R_j'. The UpdateWeights function adjusts the adaptive weights based on operator performance. The process continues until the iteration limit I_{max} is reached, and the improved routes R_j^* are returned.

- **2-OPT* (Inter-route Swap):** The 2-OPT* algorithm performs customer swaps between different mechanic routes to explore potential improvements in the objective cost, given the solution maintains feasibility. Note that this inter-route swap variant differs from traditional intra-route 2-OPT. Algorithm 11 presents the pseudocode for 2-OPT*.

The algorithm takes the initial routes R_j, the set of mechanics $\mathcal{M}$, mechanic capacity Q, and predicted priority scores $\mathcal{P}$ as input. It iterates over each pair of distinct routes (r_1, r_2) in R_j^*. For each pair of routes, it considers every possible combination of locations i from r_1 and k from r_2. The priority scores directly influence swap feasibility through the explicit condition $p_i \geq p_k$, which ensures that swaps only occur when the priority of location i is greater than or equal to that of location k, preserving the preferential treatment of high-priority machines during the optimization process.

After each swap, if the new solution R_j^* has a better objective function value $f(R_j^*)$ than the current best solution $f(R_j)$, R_j is updated with R_j^*, and the *improved* flag is set to *true*. If the swap does not improve the solution, it is immediately reverted. The objective function

Algorithm 11 2-OPT* (Inter-route Customer Swap)

Require: Initial routes R_j for cluster C_j, set of mechanics $\mathcal{M}$, mechanic capacity Q, predicted priority scores $\mathcal{P} = \{p_i \mid i \in C_j\}$
Ensure: Improved routes R_j^* for cluster C_j

$R_j^* \leftarrow R_j$, $improved \leftarrow true$
while $improved$ **do**
  $improved \leftarrow false$
  for each route pair $(r_1, r_2) \in R_j^* \times R_j^*, r_1 \neq r_2$ **do**
    for each location $i \in r_1$ **do**
      for each location $k \in r_2$ **do**
        if $p_i \geq p_k$ **and** SwapFeasible(r_1, i, r_2, k, Q) **then**
          Swap locations i and k between r_1 and r_2
          if $f(R_j^*) < f(R_j)$ **then** ▷ $f(R)$ calculates total time of routes
            $R_j \leftarrow R_j^*$, $improved \leftarrow true$
          else
            Revert the swap
          end if
        end if
      end for
    end for
  end for
end while
return R_j^*

$f(R)$ calculates the total time of the routes, including both travel and service times.

This process continues, examining all possible swaps between distinct routes, until no further improvements can be made (i.e., a full iteration occurs without any successful swaps). At this point, the algorithm terminates and returns the improved routes R_j^*.

- **Modified SOM:** Lastly, we employ a modified version of the Self-Organizing Map (SOM) algorithm tailored to solve the MDCVRP. Typically, SOMs are used for unsupervised learning, clustering, and visualization of high-dimensional data. In our approach, we adapt the SOM algorithm to optimize vehicle routes by representing each route as a circular array of neurons and introducing a decaying learning rate and neighborhood function. The circular array of neurons allows for a more natural representation of the VRP-like problem in the MDCVRP, facilitating the learning and optimization of the route sequence. The decaying learning rate and neighborhood function enable the SOM to initially explore a wide range of solutions and progressively focus on fine-tuning the best routes as the training progresses.
  Our modified SOM incorporates benchmark routing data $\mathcal{H}$ from established MDVRP instances (Oliveira, 2024) for pretraining the neural network structure. These benchmark routes serve as training patterns that help the SOM learn general routing principles and structural patterns of good solutions, using only the structural patterns (relative positions, clustering patterns) rather than absolute coordinates to avoid any data leakage. This pretraining approach enhances the algorithm's ability to generate efficient routes while respecting the specific constraints and priorities of the current problem instance. The pseudocode for the modified SOM is presented in Algorithm 12.

The algorithm initializes circular neuron arrays W_j for each depot $j \in D$ with random weights $\mathbf{w}i^j(0)$, normalizing customer and depot coordinates. Each neuron represents a potential stop in the route. The main training loop runs for Imax iterations, processing each benchmark route $\mathbf{r}$ from dataset $\mathcal{H}$. For each benchmark route, we map it to the nearest depot j based on the distance between the route's centroid

Algorithm 12 Modified Self-Organizing Maps (SOM) for MDCVRP

Require: Initial routes R_j for each depot $j \in D$, set of vehicles V, vehicle capacity M, set of customers N, predicted priority scores $\mathcal{P} = \{p_i \mid i \in N\}$, benchmark routing data $\mathcal{H}$, number of iterations $I_{\max}$, initial learning rate α_0, learning rate decay α_{decay}, initial neighborhood size σ_0, neighborhood decay σ_{decay}
Ensure: Improved routes R_j^* for all depots $j \in D$

Initialize circular arrays of neurons W_j for each depot $j \in D$ with random weights $\mathbf{w}_i^j(0)$
Normalize coordinates of customer locations and depots
for $t = 1$ to $I_{\max}$ **do**
  for each benchmark route $\mathbf{r} \in \mathcal{H}$ **do**
    Map benchmark route $\mathbf{r}$ to nearest depot j based on centroid distance
    Find the winning neuron $w = \arg\min_i |\mathbf{r} - \mathbf{w}_i^j(t)|$ in W_j
    for each neuron n in the circular array W_j **do**
      Calculate the neighborhood function $h(n, w, \sigma(t)) = \exp\left(-\frac{|n-w|^2}{2\sigma(t)^2}\right)$
      Update the weight of neuron n: $\mathbf{w}_n^j(t+1) = \mathbf{w}_n^j(t) + \alpha(t) \cdot h(n, w, \sigma(t)) \cdot (\mathbf{r} - \mathbf{w}_n^j(t))$
    end for
  end for
  Decay the learning rate: $\alpha(t+1) = \alpha(t) \cdot \exp(-\alpha_{\text{decay}})$
  Decay the neighborhood size: $\sigma(t+1) = \sigma(t) \cdot \exp(-\sigma_{\text{decay}})$
end for
for each depot $j \in D$ **do**
  $R_j^* \leftarrow$ MapRoutesToModifiedSOM$(R_j, W_j, V, M, \mathcal{P})$
end for
return R_j^* for all depots $j \in D$

and each depot location, ensuring that benchmark patterns are applied to geographically relevant regions. The winning neuron w is then determined as the neuron closest to the input route in depot array W_j, representing the best-matching stop.

Neuron weights in W_j are updated based on proximity to w using the neighborhood function $h(n, w, \sigma(t))$, which considers the current neighborhood size $\sigma(t)$. Updates shift neurons closer to the input route, guided by the learning rate $\alpha(t)$. Both $\alpha(t)$ and $\sigma(t)$ decay exponentially, allowing broad exploration initially before fine-tuning optimal routes.

The trained SOMs optimize initial routes R_j via MapRoutesToModifiedSOM, which adjusts routes based on learned patterns while ensuring feasibility with vehicle capacities M and predicted priority scores $\mathcal{P}$. The optimized routes R_j^* for all depots are returned. This modified SOM approach accounts for the multi-depot structure by maintaining separate neuron arrays per depot, associating benchmark routes with the nearest depot, and incorporating capacity constraints and priority scores to generate feasible, optimized routes.

5.4. Computational experiments

Our computational experiments utilize data from a major construction equipment service provider managing over 5000 excavators across Europe with 70 field technicians and annual service costs exceeding €6 million. To evaluate our framework's scalability and performance, we extracted representative subsets from their operations, creating test instances that reflect real-world complexity while allowing for controlled experimentation. The instances range from small regional clusters (10 locations, 2 mechanics) to district-level operations (100 locations, 10 mechanics), preserving the geographical distribution and service patterns of the full network.

To evaluate our routing model and solution approaches, we conduct computational experiments using realistic instances derived from historical data from a heavy construction equipment service company.

The goal is to assess the model's effectiveness in minimizing total time (T_{ij}) while considering predicted priority scores and worker schedules.

We generate instances based on actual geographical locations of equipment and mechanics' homes, covering small (10 locations, 2 mechanics) to large (100 locations, 10 mechanics) cases. For each category, we run ten instances, averaging metrics across 30 total instances (10 small, 10 medium, 10 large). Travel times are geocoded using Google Maps API, while service times are predicted using models from Section 4.2.

Experiments run on an Intel Core i7, 16 GB RAM, 64-bit OS, with the mathematical model implemented in IBM ILOG CPLEX 20.1 and heuristic approaches coded in Python 3.11. We evaluate and compare the performance of five solution methods across all instances.

- **Exact Method:** MILP formulation solved using a commercial solver with a 3600-second time limit.
- **Modified CFRS Variants:** Our proposed two-stage approach with three alternative intensification techniques:
  (a) Adaptive Large Neighborhood Search (ALNS)
  (b) 2-Opt* (inter-route swap)
  (c) Modified Self-Organizing Maps (SOM)
- **Baseline 1 (Current Practice):** The company's existing routing method based on historical data.
- **Baseline 2 (Greedy Heuristic):** A simple nearest-neighbor approach that assigns locations to mechanics based solely on proximity.

The key performance indicators used to evaluate the solution methods are the total time (T_{ij}) which consists of travel time plus service time, the average route position of high-priority machines ($\bar{r}_H$), and the computational time required to obtain the solutions. $\bar{r}_H$ assesses the effectiveness of the routing model in prioritizing and responding to machines with critical maintenance needs, as influenced by the priority scores used in both the mathematical formulation and the solution approach.

Let $\mathcal{H} \subseteq \mathcal{N}$ be the set of high-priority machines with predicted priority scores p_i exceeding the 75th percentile threshold τ among all machines in $\mathcal{N}$, i.e., $\mathcal{H} = \{i \in \mathcal{N} \mid p_i > \tau\}$. This threshold ensures that the top 25% of machines with the highest priority scores are classified as high-priority, aligning with the priority-based optimization in our model.

For each machine $i \in \mathcal{H}$, let $position_i(R_{jk})$ denote the position of machine i in route R_{jk} (1 for first visited, 2 for second, etc.). The average route position of high-priority machines is then calculated as:

$$\bar{r}_H = \frac{\sum_{i \in \mathcal{H}} position_i(R_{jk})}{|\mathcal{H}|}, \tag{30}$$

where $|\mathcal{H}|$ denotes the cardinality of set $\mathcal{H}$, i.e., the number of high-priority machines. A lower $\bar{r}_H$ value indicates that high-priority machines are visited earlier in their respective routes, demonstrating the effectiveness of our priority-based routing approach.

Table 8 presents the computational results across different problem sizes. The exact method achieves optimal solutions only for small instances within the 3600-second time limit. For medium and large instances marked with †, CPLEX returns the best incumbent solution found at timeout, which in these cases underperformed our heuristic methods. All CFRS variants consistently outperform both baseline methods, with CFRS (SOM) demonstrating superior performance by reducing total time by an average of 41.93% compared to Baseline 1. CFRS (ALNS) and CFRS (2-OPT*) show substantial average reductions of 38.81% and 34.05%, respectively. The improvements in high-priority machine route position ($\bar{r}_H$) are even more pronounced, with CFRS (SOM) achieving a 43.86% improvement compared to Baseline 1, meaning high-priority machines are visited much earlier in their routes. CFRS (ALNS) and CFRS (2-OPT*) deliver improvements of 40.35% and 38.60%, respectively. Paired t-tests confirmed these improvements as statistically significant ($p < 0.01$) for all CFRS variants across both performance metrics.

Among the CFRS variants, the SOM-based implementation consistently outperforms other techniques, particularly for medium and large problem instances, due to what we believe is its ability to learn spatial relationships and adapt to complex routing patterns. ALNS follows closely with strong performance through its adaptive destroy-and-repair operations, while 2-OPT* provides the most computationally efficient option with reasonable solution quality. The computational run times shows expected scaling patterns, with 2-OPT* offering the fastest execution (1.8–125.9 s across problem sizes) and SOM providing the highest solution quality at increased computational cost (60.5–628.4 s). Nonetheless, it is clear that as problem size increases, the performance advantage of CFRS variants over baseline methods becomes more pronounced.

These computational improvements have implications for the service provider's operations. Extrapolating from our test results to their network of approximately 1400 customers, the observed 41.93% reduction in total service time could free capacity across their 70 field technicians. Published benchmarks indicate that labor constitutes about 30% of construction-service costs (Liu & Ballard, 2008), unplanned downtime consumes roughly 5% (O'Brien, 2019), and premium or expedited freight represents around 5% of logistics spend (APQC, 2020). Together these cost drivers make up about 40% of the annual €6 million service budget (≈€2.4 million). Applying the 41.93% time saving to this variable portion (0.40 ×€6 000 000 × 0.4193) results in an expected annual saving of approximately €1 million, or about 17% of total service spend.

5.5. Sensitivity analysis

To assess the robustness and adaptability of our approach, we conduct a sensitivity analysis on three key factors: priority score thresholds, cluster numbers, and travel time variations.

5.5.1. Effect of priority score thresholds

We systematically vary the high-priority threshold from the 60th to 90th percentile (in 10% increments) to assess its impact on average route position ($\bar{r}_H$) and total time (T_{ij}). Fig. 6 shows that as the priority threshold increases, $\bar{r}_H$ decreases across all methods, with CFRS (SOM) consistently demonstrating superior performance.

At the 90th percentile threshold, CFRS (SOM) achieves an average position of 1.8, meaning high-priority machines are typically the first or second visited in their routes, outperforming CFRS (ALNS) by 11%, CFRS (2-OPT*) by 16%, and both baseline methods by approximately 55%. As the threshold increases, fewer machines qualify as high-priority, allowing more focused resource allocation and reduced inter-machine travel time for critical maintenance tasks.

5.5.2. Impact of priority weighting parameter (λ)

The priority weighting parameter λ in the objective function (Eq. (12)) controls the trade-off between minimizing total route time and prioritizing high-urgency maintenance tasks. To understand its impact, we varied λ from 0 to 1.0 across medium-sized instances (50 locations, 5 mechanics).

Fig. 7 shows a clear trade-off between competing objectives. As λ increases, high-priority route positions (dashed lines) decrease sharply where CFRS (SOM) achieves a 54% reduction from position 3.8 to 1.7. However, this improvement comes at the cost of increased total route times (solid lines), which rise by 17% as mechanics deviate from geographically optimal paths to serve urgent equipment first.

The shaded region (λ = 0.3–0.5) shows the operational sweet spot. Within this range, organizations can reduce high-priority route positions by 38%–42% while increasing total route time by only 2%–5%.

Table 8
Computational results for different problem sizes.

Metric	Method	Small (10 × 2)	Medium (50 × 5)	Large (100 × 10)	Avg. improvement (%)[b]
T_{ij} (min)	Exact	108.6	493.2†	1893.2†	16.45
	CFRS (ALNS)	123.2	459.1	927.3	38.81
	CFRS (2-OPT*)	131.8	489.6	985.8	34.05
	CFRS (SOM)	116.4	433.2	890.5	41.93
	Baseline 1	203.3	767.5	1423.7	–
	Baseline 2	185.7	723.1	1274.3	8.31
$\bar{r}_H$ (position)	Exact	1.3	2.2†	3.2†	47.37
	CFRS (ALNS)	1.5	2.4	3.8	40.35
	CFRS (2-OPT*)	1.6	2.5	3.9	38.60
	CFRS (SOM)	1.4	2.3	3.5	43.86
	Baseline 1	2.5	4.1	6.3	–
	Baseline 2	2.3	3.9	5.8	8.77
Computational time (s)	Exact	15.2	3600.0†	3600.0[a]	–
	CFRS (ALNS)	15.3	104.6	424.7	–
	CFRS (2-OPT*)	1.8	28.2	125.9	–
	CFRS (SOM)	60.5	207.8	628.4	–
	Baseline 1	–	–	–	–
	Baseline 2	1.2	12.6	45.3	–

[a] Time limit reached; best incumbent solution reported (not optimal).

[b] $Avg\ improvement = \frac{\sum_{i=1}^{n}\left(\frac{\text{Baseline}_i - \text{Method}_i}{\text{Baseline}_i}\right)\times 100}{n}$.

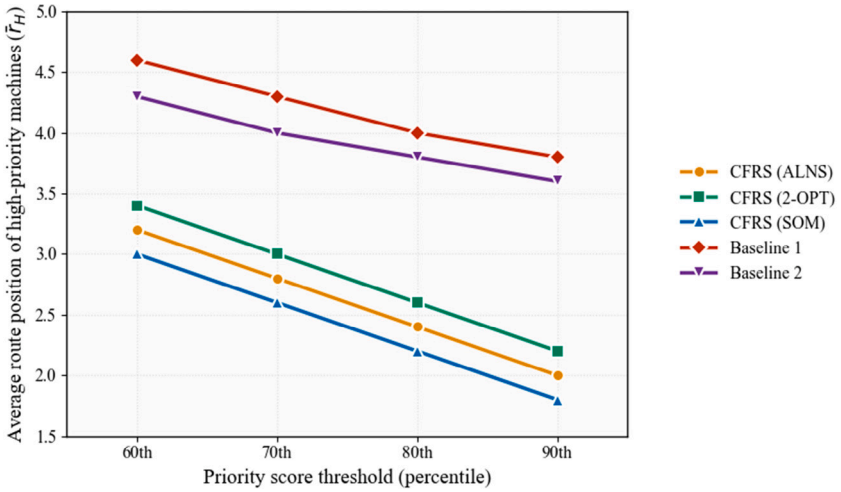

Fig. 6. Effect of priority score threshold on average route position of high-priority machines ($\bar{r}_H$).

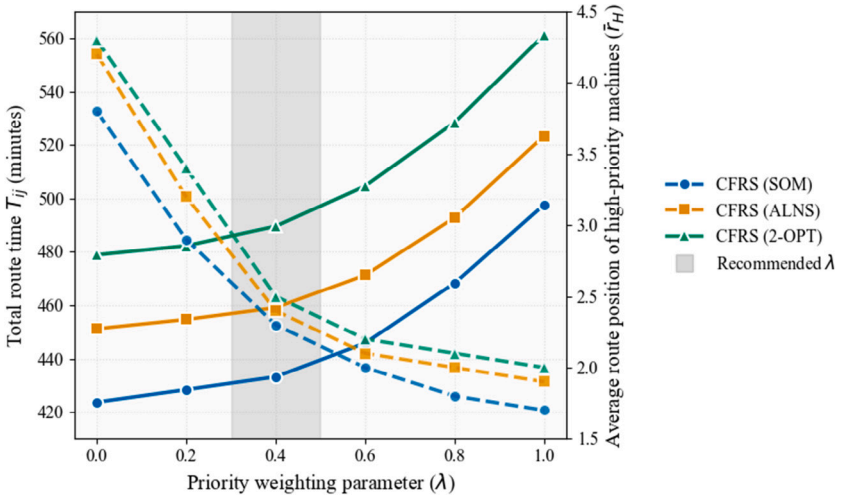

Fig. 7. Impact of priority weighting parameter (λ) on routing performance. Solid lines represent total route time T_{ij} (left axis) while dashed lines show average route position of high-priority machines $\bar{r}_H$ (right axis). The shaded region indicates the recommended operational range.

Table 9
Performance metrics for CFRS variants under different cluster numbers (percentage of clusters = number of clusters/number of locations).

Method	Clusters (%)	T_{ij} (min)	$\bar{r}_H$ (position)	Comp. time (s)
CFRS (ALNS)	10	958	3.9	407
	20	928	3.8	452
	30	907	3.7	523
CFRS (2-OPT*)	10	988	4.1	129
	20	967	3.9	142
	30	941	3.8	164
CFRS (SOM)	10	913	3.6	606
	20	892	3.5	662
	30	887	3.4	709

Beyond $\lambda = 0.5$, the rate of efficiency loss accelerates without commensurate gains in route position, indicating diminishing returns. Nonetheless, CFRS (SOM) consistently outperforms the alternatives across all λ values. At $\lambda = 0.4$ (used in our main experiments), it achieves a total time of 433.2 min and priority position of 2.3, approximately 6%–12% better than ALNS and 2-OPT*.

5.5.3. Impact of cluster numbers

Next, we analyzed how partitioning machines into a different number of clusters affects performance. Table 9 presents performance metrics when varying the percentage of clusters from 10% to 30% of total service locations (e.g., for 50 locations, 10% = 5 clusters, 20% = 10 clusters, 30% = 15 clusters). Larger numbers of clusters create smaller subproblems that are easier to optimize individually, improving both T_{ij} and $\bar{r}_H$. For example, increasing cluster count from 10% to 30% allowed CFRS (SOM) to reduce total route times by nearly 3% and high-priority route positions by a similar margin.

However, this gain in solution quality comes at the expense of higher computational overhead, as the optimization procedure must allocate time to refine a growing number of clusters. Organizations with frequent, real-time scheduling needs may favor fewer clusters to reduce solution times, whereas those emphasizing route quality or serving especially critical machinery may accept slower computation to gain improved routing outcomes.

Nevertheless, relying on an overly strict threshold may leave moderately important tasks underserved. Organizations must therefore choose thresholds that align with service-level objectives, balancing the need for swift attention to truly critical repairs against the risk of neglecting intermediate-priority machines.

5.5.4. Robustness to travel time variations

To simulate real-world uncertainties such as traffic or weather conditions, we introduced random perturbations in travel times of ±10% and ±20%. We then evaluated how well each method performs under these variations.

Fig. 8 illustrates that while all methods experience increased variability in performance as travel time uncertainty increases, the CFRS variants, particularly CFRS (ALNS) and CFRS (SOM), maintain more consistent performance compared to the baseline methods. At 20% travel time variation, the interquartile range for CFRS (SOM) is approximately 30 min, compared to 50 min for Baseline 1 and 40 min for Baseline 2.

The robustness of CFRS variants to travel time variations can be attributed to their adaptive nature and the integration of priority scores in routing decisions. The ALNS algorithm's destroy and repair operations allow continuous exploration of diverse solutions, making it more resilient to travel time changes. It adapts its routing decisions quickly based on the new travel times, maintaining solution quality under uncertainty.

More specifically, SOM's robustness stems from its ability to learn and generalize routing patterns. As travel times vary, it can apply these learned patterns to maintain good performance, such as grouping nearby high-priority machines, a strategy that remains effective even as specific travel times change. The priority-based routing ensures that high-priority machines are served promptly, minimizing the impact of travel time variations on the most critical tasks.

The 2-OPT* variant, while outperforming baseline methods, is more sensitive to travel time variations compared to ALNS and SOM. This is due to its local search nature, which may struggle to escape local optima with travel time changes, lacking the global exploration of ALNS or the pattern learning of SOM.

6. Discussion

This section examines the practical implications of our framework for the construction industry. We first present managerial insights and implementation strategies, including considerations for real-time deployment and organizational change management. We then discuss the limitations of our current approach and identify promising directions for future research.

6.1. Managerial insights and implementation strategy

The deployment of our framework requires careful consideration of both technical and organizational factors. Construction companies typically allocate only 1%–2% of annual revenue to technology investments (Blanco et al., 2023), yet industry research indicates that firms achieving full digital transformation realize average productivity gains of 14%–15% and cost reductions of 4%–6% (Koeleman et al., 2019). For our maintenance-optimization framework, the return on investment can be particularly compelling given that poor data management alone cost the construction industry an estimated $1.85 trillion globally in 2020 (Autodesk Inc. & FMI Corporation, 2021).

The integrated nature of our framework (Fig. 1) creates value through component synergies that traditional siloed approaches miss. The flow from predictive modeling to priority scoring to routing optimization means that improvements in any component automatically enhance overall system performance. For instance, when our SMU predictions identify equipment approaching critical maintenance thresholds, this information immediately updates priority scores, which in turn influences routing decisions. This automation reduces the administrative burden while ensuring maintenance decisions remain data-driven and consistent with organizational objectives.

We recommend a phased implementation approach to minimize disruption and ensure sustainable adoption. Phase 1 (months 1–3) should focus on deploying the priority-scoring system using existing maintenance records and establishing baseline performance metrics. Phase 2 (months 4–6) introduces predictive modeling for labor costs, service durations, and spare-parts forecasting, leveraging historical data already captured in enterprise systems. Phase 3 (months 7–9) integrates routing optimization, beginning with the computationally efficient 2-OPT* algorithm to demonstrate quick wins before transitioning to more sophisticated approaches. Finally, Phase 4 (months 10–12) implements full system integration including SOM-based optimization for maximum performance gains. A tiered roll-out of this kind is consistent with empirical evidence that phased implementation, coupled with early

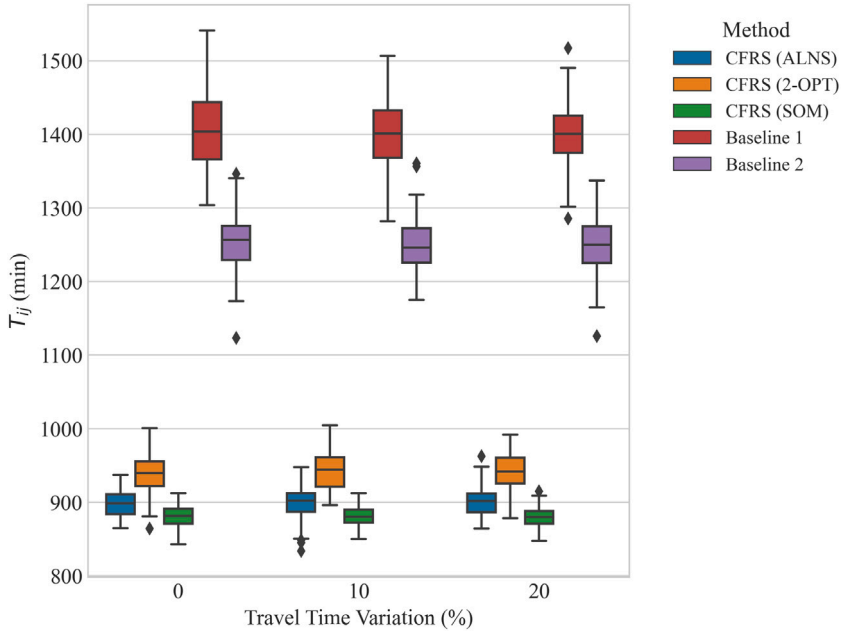

Fig. 8. Effect of travel time variation on total time (T_{ij}).

wins, builds momentum and confidence in construction-technology projects (Lourenço et al., 2025).

Change management represents a critical success factor, as research indicates that 72% of construction firms still rely on paper-based processes despite available digital alternatives (Bluebeam Inc., 2024). This resistance stems from psychological factors including status-quo bias and the comfort of familiar workflows (Maali et al., 2020). To address these barriers, organizations should implement training programs that show tangible benefits to frontline workers, such as reduced overtime through better scheduling and decreased emergency call-outs. Creating internal champions who can advocate for the system and providing continuous support during the transition period are essential for overcoming the human elements of technological change (Lines et al., 2015).

For organizations requiring real-time or near-real-time scheduling capabilities, our framework offers several adaptation strategies. While the SOM-based approach provides superior solution quality, its computational requirements (60.5–628.4 s across problem sizes) may limit immediate deployment for frequent rescheduling. We propose a hybrid approach where the 2-OPT* algorithm generates rapid initial solutions for urgent scheduling needs, with SOM refinement occurring during low-demand periods. Additionally, maintaining a library of pre-trained SOM models for common operational scenarios can help reduce response times for typical rescheduling requests. Organizations can further enhance real-time performance through parallel computing architectures or by developing lightweight machine-learning surrogates trained on historical SOM solutions.

6.2. Limitations and future research

While our framework demonstrates considerable improvements in maintenance optimization, several limitations warrant consideration. First, our analysis focused exclusively on excavators within a single company's operations. Construction fleets typically comprise diverse equipment types, bulldozers, cranes, loaders, each with distinct maintenance requirements and operational patterns. Extending our framework to multi-equipment scenarios would require equipment-specific modifications to the priority scoring components and predictive models, particularly for the APS calculations where different equipment types may have varying optimal maintenance intervals.

Second, our models assume deterministic travel and service times, yet construction environments are inherently uncertain. Weather conditions, traffic patterns, and unexpected equipment failures can significantly impact scheduling reliability. Future research should incorporate stochastic optimization techniques or robust optimization approaches that explicitly account for uncertainty in both travel times and service durations. Scenario-based planning or Monte Carlo simulations could provide more resilient scheduling solutions that maintain performance under various operational conditions.

The environmental dimension presents another avenue for enhancement. As construction companies face mounting pressure to reduce carbon emissions, incorporating fuel consumption and emissions metrics into the routing optimization could transform our single-objective model into a multi-objective framework. This extension would require balancing traditional efficiency metrics with environmental impact, potentially using Pareto optimization techniques to identify solutions that optimize across multiple sustainability dimensions.

Finally, our current framework does not account for technician skill specialization, assuming all mechanics can service any equipment type. In practice, certain maintenance tasks require specialized certifications or expertise. Future iterations should incorporate skill-based assignment constraints, potentially using mixed-integer programming formulations that match technician capabilities with equipment requirements while maintaining routing efficiency.

7. Conclusion

This study advances heavy-equipment maintenance optimization by integrating three key pillars: priority scoring, predictive modeling, and routing heuristics. Our approach leverages routinely collected operational data to achieve a 64.4% reduction in prediction error versus a naïve current-SMU baseline (PMI = −0.644) based on our ten-machine evaluation set without requiring specialized sensors, while predictive models achieve MAPE of 4.24% (labor cost) and 9.32% (service duration), outperforming company baselines. The routing framework, combining a modified CFRS strategy with metaheuristics (ALNS, 2-OPT*, and a novel SOM adaptation), delivers substantial operational gains, reducing total service time by 41.93% and improving high-priority route position by 43.86% compared to baseline.

These results demonstrate that maintenance optimization is achievable without specialized IoT infrastructure, making the framework accessible to typical construction firms. Sensitivity analyses also confirm robustness across varying operational conditions, from cluster granularity to travel-time uncertainty. Looking ahead, the model can be extended to accommodate multiple equipment types, real-time data streams, and skill-based labor assignments. Incorporating environmental metrics like fuel usage or emissions could also transform the approach into a multi-objective tool that balances operational and sustainability goals. As the construction industry tries to embrace digitalization, our approach provides a foundation for adaptive, data-driven maintenance systems, paving the way for more efficient and sustainable infrastructure development.

CRediT authorship contribution statement

Lais Wehbi: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Sandjai Bhulai:** Writing – review & editing, Supervision, Methodology, Formal analysis, Conceptualization. **Jesper Slik:** Writing – review & editing, Validation, Supervision, Resources, Project administration, Data curation.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the revision of this work, the authors used an LLM (Claude) for language editing. All study design, analysis, interpretation of results, and conclusions are the authors' own work. The authors take full responsibility for the content of this manuscript.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors gratefully acknowledge Bram van der Meij for his support in providing business insights, practical validation, and data for this research. The authors also thank Mathijs Pellemans and Yasmin Roshandel for their valuable discussions and constructive feedback that helped improve this work.

Data availability

The data that has been used is confidential.

References

Abioye, S. O., Oyedele, L. O., Akanbi, L., Ajayi, A., Davila Delgado, J. M., Bilal, M., Akinade, O. O., & Ahmed, A. (2021). Artificial intelligence in the construction industry: A review of present status, opportunities and future challenges. *Journal of Building Engineering*, *44*, Article 103299. http://dx.doi.org/10.1016/j.jobe.2021.103299, URL: https://www.sciencedirect.com/science/article/pii/S2352710221011578.

Amrina, E., & Aridharma, D. (2016). Sustainable maintenance performance evaluation model for cement industry. In *2016 IEEE international conference on industrial engineering and engineering management* (pp. 350–354). http://dx.doi.org/10.1109/IEEM.2016.7797895.

APQC (2020). Metric of the month: Mitigating expedited costs in logistics. Supply & Demand Chain Executive. URL: https://www.sdcexec.com/transportation/article/21116936/apqc-metric-of-the-month-mitigating-expedited-costs-in-logistics. (Accessed 30 July 2025).

Armstrong, W., Parsons, J., Terisno, L., & Thibert, J. (2024). More uptime, lower cost: Boosting organizational health in maintenance. https://www.mckinsey.com/capabilities/operations/our-insights/more-uptime-lower-cost-boosting-organizational-health-in-maintenance. (Accessed 04 August 2024).

Aune, M., Winther, M., Dore, C., & Lambrecht, U. (2020). Regionalized environmental impacts of construction machinery. *The International Journal of Life Cycle Assessment*, *25*, 1584–1596. http://dx.doi.org/10.1007/s11367-020-01769-x.

Autodesk Inc., & FMI Corporation (2021). *Harnessing the data advantage in construction: Technical Report*, Autodesk Construction Cloud, URL: https://www.autodesk.com/blogs/construction/autodesk-fmi-study-global-construction-industry-data-strategies/.

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, *13*(2).

Bevilacqua, M., & Braglia, M. (2000). The analytic hierarchy process applied to maintenance strategy selection. *Reliability Engineering & System Safety*, *70*(1), 71–83.

Biteus, J., & Lindgren, T. (2017). Planning flexible maintenance for heavy trucks using machine learning models, constraint programming, and route optimization. *SAE International Journal of Materials and Manufacturing*, *10*(3), 306–315.

Blanco, J. L., Rockhill, D., Sanghvi, A., & Torres, A. (2023). From start-up to scale-up: Accelerating growth in construction technology. *McKinsey & Company Insights*, Article published 3 May 2023. URL: https://www.mckinsey.com/industries/private-capital/our-insights/from-start-up-to-scale-up-accelerating-growth-in-construction-technology.

Bluebeam Inc. (2024). *Building the future: Bluebeam AEC technology outlook 2025: Technical Report*, Bluebeam Global Press, Press release, 6 Nov 2024. URL: https://press.bluebeam.com/2024/11/survey-finds-ai-has-taken-hold-in-aec/.

Box, G. E., & Jenkins, G. M. (1970). *Time series analysis: forecasting and control*. Holden-Day.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Brockwell, P. J., & Davis, R. A. (2002). *Introduction to time series and forecasting*. Springer.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785–794).

Chen, H. P., & Ying, K. C. (2022). Artificial intelligence in the construction industry: Main development trajectories and future outlook. *Applied Sciences*, *12*(12), http://dx.doi.org/10.3390/app12125832, URL: https://www.mdpi.com/2076-3417/12/12/5832.

Cheng, J. C., Chen, W., Chen, K., & Wang, Q. (2020). Data-driven predictive maintenance planning framework for mep components based on BIM and IoT using machine learning algorithms. *Automation in Construction*, *112*, Article 103087. http://dx.doi.org/10.1016/j.autcon.2020.103087.

Close, T., & Tideswell, S. (2012). Planning to fix: Improving maintenance efficiency. https://www.mckinsey.com/capabilities/operations/our-insights/planning-to-fix-improving-maintenance-efficiency. (Accessed 04 August 2024).

Coleman, C., Damodaran, S., Chandramouli, M., & Deuel, E. (2017). Making maintenance smarter: Predictive maintenance and the digital supply network. https://www2.deloitte.com/us/en/insights/focus/industry-4-0/using-predictive-technologies-for-asset-maintenance.html.

da Silva, R. F., Bellinello, M. M., de Souza, G. F. M., Antomarioni, S., Bevilacqua, M., & Ciarapica, F. E. (2021). Deciding a multicriteria decision-making (MCDM) method to prioritize maintenance work orders of hydroelectric power plants. *Energies*, *14*(24), http://dx.doi.org/10.3390/en14248281, URL: https://www.mdpi.com/1996-1073/14/24/8281.

Dekker, R. (1996). Applications of maintenance optimization models: a review and analysis. *Reliability Engineering & System Safety*, *51*(3), 229–240.

Dickey, D. A., & Fuller, W. A. (1979). Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American Statistical Association*, *74*(366a), 427–431.

Green, C. (2023). The transformative impact of preventive maintenance in construction. https://www.maintworld.com/Asset-Management/The-Transformative-Impact-of-Preventive-Maintenance-in-Construction.

Harikrishnakumar, R., Nannapaneni, S., Nguyen, N. H., Steck, J. E., & Behrman, E. C. (2020). A quantum annealing approach for dynamic multi-depot capacitated vehicle routing problem. arXiv preprint arXiv:2005.12478.

Hicks, G. (2019). Equipment downtime is costing your workplace. https://www.iofficecorp.com/blog/equipment-downtime. (Accessed 04 August 2024).

Hill, A. (2021). World's largest manufacturers lose almost $1 trillion a year to machine failures. https://www.automation.com/en-us/articles/june-2021/world-largest-manufacturers-lose-almost-1-trillion. (Accessed 04 August 2024).

Hilti Corporation (2023). Hilti sustainability report 2023. URL: https://www.hilti.group/content/dam/documents/Media-Release/company-report/hilti-sustainability-report.pdf. (Accessed 05 August 2024).

Him, L. C., Poh, Y. Y., & Pheng, L. W. (2019). IoT-based predictive maintenance for smart manufacturing systems. In *Proceedings of APSIPA annual summit and conference* (pp. 1942–1944). Lanzhou, China: http://dx.doi.org/10.1109/APSIPAASC47483.2019.9023187.

Hocking, R. R. (1976). A biometrics invited paper. The analysis and selection of variables in linear regression. *Biometrics*, 1–49.

Hong, B., & Lü, L. (2022). Assessment of emissions and energy consumption for construction machinery in earthwork activities by incorporating real-world measurement and discrete-event simulation. *Sustainability*, *14*, 5326. http://dx.doi.org/10.3390/su14095326.

Hovnanian, G., Kroll, K., & Sjödin, E. (2019). How analytics can drive smarter engineering and construction decisions. https://www.mckinsey.com/capabilities/operations/our-insights/how-analytics-can-drive-smarter-engineering-and-construction-decisions.

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

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, *22*(4), 679–688.

Januschowski, T., Gasthaus, J., Wang, Y., Salinas, D., Flunkert, V., Bohlke-Schneider, M., & Callot, L. (2020). Criteria for classifying forecasting methods. *International Journal of Forecasting*, *36*(1), 167–177.

Jarque, C. M., & Bera, A. K. (1980). Efficient tests for normality, homoscedasticity and serial independence of regression residuals. *Economics Letters*, *6*(3), 255–259.

Jbili, S., Chelbi, A., Radhoui, M., & Kessentini, M. (2018). Integrated strategy of vehicle routing and maintenance. *Reliability Engineering & System Safety*, *170*, 202–214. http://dx.doi.org/10.1016/j.ress.2017.09.030, URL: https://www.sciencedirect.com/science/article/pii/S0951832017303174.

Joe Opoku, D.-G., & Zeeshan, A. (2021). A digital twin for cybersecurity in building information modelling (BIM). *Computers in Industry*, *129*, Article 103450. http://dx.doi.org/10.1016/j.compind.2021.103450.

Koeleman, J., Ribeirinho, M. J., Rockhill, D., Sjödin, E., & Strube, G. (2019). Decoding digital transformation in construction. *McKinsey & Company Insights*, Research by the McKinsey Global Institute. URL: https://www.mckinsey.com/capabilities/operations/our-insights/decoding-digital-transformation-in-construction.

Kwiatkowski, D., Phillips, P. C., Schmidt, P., & Shin, Y. (1992). Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? *Journal of Econometrics*, *54*(1–3), 159–178.

Lines, B. C., Sullivan, K. T., Smithwick, J. B., & Mischung, J. (2015). Overcoming resistance to change in engineering and construction: Change management factors for owner organizations. *International Journal of Project Management*, *33*(5), 1170–1179. http://dx.doi.org/10.1016/j.ijproman.2015.01.009.

Liu, M., & Ballard, G. (2008). Improving labor productivity through production control. In *Proceedings of the 16th annual conference of the international group for lean construction* (pp. 657–666). Manchester, UK: Cites Harmon & Cole (2006) for the 30–50% labour-cost share. URL: http://www.iglc.net/Papers/Details/592.

Ljung, G. M., & Box, G. E. (1978). On a measure of lack of fit in time series models. *Biometrika*, *65*(2), 297–303.

López-Santana, E., Akhavan-Tabatabaei, R., Dieulle, L., Labadie, N., & Medaglia, A. L. (2016). On the combined maintenance and routing optimization problem. *Reliability Engineering & System Safety*, *145*, 199–214.

López-Santana, E., Méndez, G., & Franco, C. (2023). On the multi-period combined maintenance and routing optimisation problem. *International Journal of Production Research*, *61*(23), 8265–8290. http://dx.doi.org/10.1080/00207543.2023.2180301.

Lourenço, M. P., Arantes, A., & Costa, A. A. (2025). Barriers to building information modeling (BIM) implementation in late-adopting EU countries: The case of Portugal. *Buildings*, *15*(10), 1651. http://dx.doi.org/10.3390/buildings15101651.

Maali, O., Lines, B., Smithwick, J., Hurtado, K., & Sullivan, K. (2020). Change management practices for adopting new technologies in the design and construction industry. *Journal of Information Technology in Construction*, *25*, 325–341. http://dx.doi.org/10.36680/J.ITCON.2020.019.

Milan, A., Rezatofighi, S., Garg, R., Dick, A., & Reid, I. (2017). Data-driven approximations to NP-hard problems. In S. Singh, & S. Markovitch (Eds.), *Proceedings of the thirty-first AAAI conference on artificial intelligence* (pp. 1453–1459). United States of America: Association for the Advancement of Artificial Intelligence (AAAI), AAAI Conference on Artificial Intelligence 2017, AAAI 2017 ; Conference date: 04-02-2017 Through 10-02-2017. URL: http://www.aaai.org/Conferences/AAAI/aaai17.php.

Montoya-Torres, J. R., López Franco, J., Nieto Isaza, S., Felizzola Jiménez, H., & Herazo-Padilla, N. (2015). A literature review on the vehicle routing problem with multiple depots. *Computers & Industrial Engineering*, *79*, 115–129. http://dx.doi.org/10.1016/j.cie.2014.10.029, URL: https://www.sciencedirect.com/science/article/pii/S036083521400360X.

Muchiri, P., Pintelon, L., Gelders, L., & Martin, H. (2011). Development of maintenance function performance measurement framework and indicators. *International Journal of Production Economics*, *131*(1), 295–302.

Niu, Y., Lu, W., Chen, K., Huang, G. G., & Anumba, C. (2016). Smart construction objects. *Journal of Computing in Civil Engineering*, *30*(4), Article 04015070. http://dx.doi.org/10.1061/(ASCE)CP.1943-5487.0000550, URL: https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29CP.1943-5487.0000550.

Niu, Y., Lu, W., Liu, D., Chen, K., Anumba, C., & Huang, G. G. (2017). An SCO-enabled logistics and supply chain–management system in construction. *Journal of Construction Engineering and Management*, *143*(3), Article 04016103. http://dx.doi.org/10.1061/(ASCE)CO.1943-7862.0001232, URL: https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29CO.1943-7862.0001232.

O'Brien, L. (2019). Key performance metrics and indicators survey highlights industry issues. *InTech*, ARC Advisory Group survey reports unscheduled downtime equal to 5% of total output. URL: https://www.isa.org/intech-home/2019/march-april/features/key-performance-metrics-and-indicators-survey-high.

Oliveira, F. B. (2024). MDVRP-instances. Multiple Depot VRP Instances, https://github.com/fboliveira/MDVRP-Instances.

Sivanuja, T., Udawatta, N., Karunasena, G., & Al-Ameri, R. (2024). Barriers to adopting digital technologies to implement circular economy practices in the construction industry: A systematic literature review. *Sustainability*, *16*, 3185. http://dx.doi.org/10.3390/su16083185.

Taylor, S. J., & Letham, B. (2018). Forecasting at scale. *The American Statistician*, *72*(1), 37–45.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B. Statistical Methodology*, *58*(1), 267–288.

Toru, E., & Yilmaz, G. (2023). A multi-depot vehicle routing problem with time windows for daily planned maintenance and repair service planning. *Pamukkale University Journal of Engineering Sciences*, *29*(8), 913–919. http://dx.doi.org/10.5505/pajes.2023.99569.

Tukey, J. W. (1977). *Exploratory data analysis*: *vol. 2*, Reading, Mass..

UN Environment Programme (2024). Not yet built for purpose: Global building sector emissions still high and rising. https://www.unep.org/news-and-stories/press-release/not-yet-built-purpose-global-building-sector-emissions-still-high.

United Nations Environment Programme (2019). *2019 Global status report for buildings and construction*: *Technical Report*, UNEP, URL: https://www.unenvironment.org/resources/report/2019-global-status-report-buildings-and-construction.

Wang, L., Li, B., & Zhao, X. (2024). Multi-objective predictive maintenance scheduling models integrating remaining useful life prediction and maintenance decisions. *Computers & Industrial Engineering*, *197*, Article 110581. http://dx.doi.org/10.1016/j.cie.2024.110581, URL: https://www.sciencedirect.com/science/article/pii/S0360835224007022.

Zhang, K., He, F., Zhang, Z., Lin, X., & Li, M. (2020). Multi-vehicle routing problems with soft time windows: A multi-agent reinforcement learning approach. *Transportation Research Part C: Emerging Technologies*, *121*, Article 102861. http://dx.doi.org/10.1016/j.trc.2020.102861, URL: https://www.sciencedirect.com/science/article/pii/S0968090X20307610.

Zhao, L., Zhu, Y., & Zhao, T. (2022). Deep learning-based remaining useful life prediction method with transformer module and random forest. *Mathematics*, *10*(16), 2921. http://dx.doi.org/10.3390/math10162921.