Adaptive Warehouse Task Allocation Using Stacked Surrogate Models and Active Learning with Simulation-Based Validation

Huijuan Hao[1*], Dezhi Kong[2], Kunfeng Liu[1]
[1]Center of Modern Educational Technology, Hebei University of Water Resources and Electric Engineering, Cangzhou 061001, Hebei, China
[2]Basic Department, Hebei University of Water Resources and Electric Engineering, Cangzhou 061001, Hebei, China
E-mail: haohuijuan@hhwe.edu.cn
*Corresponding author

Keywords: Warehouse management, surrogate modelling, stacked ensemble, attention-based neural network, LightGBM, active learning, simulation validation

Received: January 15, 2026

Modern warehouse operations face increasing order volumes, volatile demand, and shrinking lead times, challenging conventional static task dispatching policies such as FIFO or priority-based picking. This study proposes a closed-loop intelligent warehouse management framework that integrates task-level feature engineering, lightweight policy simulation, stacked surrogate modelling, and active learning retraining to enable adaptive, real-time decision support. The surrogate model combines an attention-enhanced TensorFlow multilayer perceptron, a LightGBM classifier, and a logistic regression meta-learner to predict optimal dispatching policies for dynamically generated warehouse tasks. Experiments were conducted using a publicly available logistics warehouse dataset comprising over 3,000 inventory records, from which realistic task batches were generated and evaluated across multiple dispatching strategies. Simulation-based validation was performed using a SimPy discrete-event model capturing contention between automated guided vehicles and human pickers under varying workload and resource conditions. The stacked surrogate achieved validation accuracy exceeding 98% with balanced precision and recall across all policy classes. Simulation results showed stable throughput, reduced average waiting times, and balanced resource utilisation compared with FIFO baselines. Additional robustness analyses demonstrated consistent model behaviour under demand variability and resource disturbances. The integration of an active learning loop enables continuous recalibration of the surrogate model using observed simulation KPIs, ensuring adaptability to changing operational contexts. The proposed framework provides a computationally efficient and interpretable approach for adaptive warehouse task allocation and demonstrates broader applicability to intelligent decision-making in cyber-physical systems aligned with Industry 4.0 principles.

Povzetek: Študija predstavlja inteligenten, prilagodljiv sistem za upravljanje skladišč, ki z uporabo simulacij in strojnega učenja optimizira razporejanje nalog ter izboljšuje učinkovitost in odzivnost v dinamičnih razmerah.

1 Introduction

Warehouse management systems are undergoing rapid transformation under the pressures of increasing order volumes, shrinking delivery lead times, and volatile customer demand patterns. Conventional task dispatching and resource allocation strategies such as first-in-first-out (FIFO), nearest-task assignment, or priority-based picking remain widely adopted in industry due to their simplicity. However, these static, rule-driven approaches fail to accommodate the multi-objective nature of modern warehouse operations, where throughput, task completion time, waiting time, and equipment utilisation must be optimised simultaneously. This mismatch often leads to bottlenecks, underutilised resources, and suboptimal service levels, particularly in high-variability environments such as e-commerce fulfilment centres. To address these *Task Allocation* challenges, simulation-based optimisation has emerged as a powerful decision-support tool. Platforms such as FlexSim and AnyLogic allow

researchers and practitioners to test alternative warehouse conFigureurations and policies under realistic operating conditions. While such tools provide valuable insight, their computational cost poses a significant barrier to real-time decision-making. Executing full-scale discrete-event simulations for every new operational scenario is prohibitively time-consuming, making them unsuitable for adaptive, online control. A promising alternative lies in the adoption of surrogate modelling techniques. Surrogate models are data-driven approximators of simulation outputs that can rapidly estimate the performance of candidate policies while preserving the key trade-offs observed in detailed simulations. By reducing reliance on computationally expensive simulators, surrogate models enable near-instantaneous policy evaluation, thereby bridging the gap between offline planning and real-time decision support. In this study, a closed-loop intelligent warehouse visualisation and management framework inspired by FlexSim is proposed but designed for computational efficiency and adaptability. The framework integrates five key components, Feature engineering and task generation, where rich item-level features (demand, costs, travel distance, stock-risk indicators) are aggregated into task-level descriptors, Policy simulation and labelling, where candidate dispatching policies (nearest-first, priority-based, batch-optimised, and balanced) are scored on heuristic KPIs, producing labeled data for supervised training, Surrogate learning, using a stacked ensemble model that combines a TensorFlow-based attention MLP, LightGBM, and a logistic regression meta-learner to predict the best policy for each task, Simulation integration, where recommendations are validated in a SimPy-based WarehouseSim, modelling contention between automated guided vehicles (AGVs) and human pickers and Active learning retraining, in which real simulation outcomes (throughput, waiting time, utilisation) are reintegrated into the surrogate, continuously improving its predictive fidelity. This hybrid approach offers several advantages over prior work like it overcomes the rigidity of rule-based policies by enabling adaptive, data-driven recommendations, it significantly reduces computational overhead compared to traditional discrete-event simulation, while maintaining strong alignment with true system dynamics and it achieves high predictive accuracy: the stacked surrogate reached a validation accuracy of 99.25%, with precision and recall consistently near 1.0 across all policy classes. Finally, integration with active learning ensures that the model does not remain static but instead continuously adapts to new operational contexts, maintaining robustness in dynamic environments. The preliminary experiments demonstrate that surrogate-guided policy recommendations improve throughput and reduce waiting times relative to FIFO baselines, while achieving near-perfect agreement with full simulation outputs. The proposed framework therefore represents a computationally efficient, accurate, and adaptive solution for intelligent warehouse management, aligning with the broader goals of Industry 4.0 and smart logistics systems.

1.1 Problem statement

Modern warehouses are expected to handle increasingly complex and high-volume order fulfillment processes while meeting strict service-level agreements. Existing rule-based dispatching strategies such as FIFO, nearest-task, or simple priority allocation are unable to balance competing objectives like throughput, waiting time, and equipment utilisation under dynamic operational conditions. While discrete-event simulation platforms like FlexSim, AnyLogic allow exploration of alternative warehouse strategies, their high computational costs make them unsuitable for real-time decision support or online adaptive control. The central problem, therefore, lies in developing a computationally efficient, adaptive, and accurate framework that can both recommend optimal task dispatching policies and continuously refine its decision-making capabilities without relying exclusively on time-consuming simulations.

1.2 Research gap

Despite significant progress in simulation-based warehouse management research, few gaps remain unaddressed like High Computational Overhead of Simulation Models – Traditional discrete-event simulators (FlexSim, AnyLogic) require extensive time and resources, preventing their use in real-time or online decision-making environments, Limited Adaptability of Rule-Based Policies – Widely used approaches such as FIFO or nearest-first assignment cannot adapt to multi-objective trade-offs (throughput vs. wait time vs. utilisation), leading to suboptimal performance in dynamic warehouse conditions, Scarcity of Real-Time and Contextual Data – Many warehouse studies rely on static or synthetic datasets, limiting their applicability to real-world, real-time environments where task arrivals, resource availability, and demand patterns fluctuate unpredictably, Lack of Closed-Loop Learning – Existing works typically perform one-time comparisons of policies but do not integrate active learning or feedback loops where simulation outcomes continuously refine surrogate models. This leads to performance degradation as operational contexts evolve, and Challenges in Generalisability – Current surrogate models are often

designed for narrow scenarios or limited feature sets, and fail to capture bi-directional resource-task interactions such as AGV–picker contention. Thus, there exists a pressing need for a framework that addresses these gaps by combining data-driven surrogate modelling, lightweight simulation, and active learning to enable adaptive, accurate, and real-time warehouse policy recommendations.

Research objectives and research questions

The primary objective of this study is to develop a computationally efficient and adaptive framework for warehouse task allocation that integrates data-driven surrogate modelling with lightweight simulation and active learning. The proposed system aims to support real-time decision-making while maintaining interpretability and operational robustness.

To achieve this objective, the study addresses the following research questions:

RQ1: Can a stacked surrogate model accurately predict optimal warehouse task dispatching policies based on engineered task-level features?

RQ2: How effectively can lightweight simulation be used to validate surrogate-driven policy recommendations under realistic warehouse conditions?

RQ3: Does the integration of an active learning feedback loop improve the adaptability and long-term reliability of surrogate models in dynamic operational environments?

RQ4: How robust is the proposed framework under variations in demand, resource availability, and stochastic operational conditions?

By systematically addressing these questions, the study seeks to demonstrate that a closed-loop surrogate-learning approach can provide accurate, interpretable, and adaptive decision support for modern warehouse management systems.

1.3 Contributions of the study

This study makes the following key contributions,

- Proposes a Closed-Loop Intelligent Warehouse Management Framework that integrates task-level feature engineering, surrogate modelling, simulation validation, and active learning retraining, inspired by FlexSim but optimised for computational efficiency.
- Develops a Stacked Surrogate Model combining TensorFlow attention-based MLP, LightGBM, and logistic regression meta-learner, achieving 99.25% validation accuracy with near-perfect precision and recall across all dispatching policies.
- Introduces a Lightweight SimPy-Based Warehouse Simulation that captures resource contention between AGVs and pickers, providing realistic performance metrics (throughput, wait time, utilisation) without the computational burden of full-scale discrete-event simulators.
- Implements an Active Learning Loop, where simulation outcomes are fed back into the surrogate model, enabling continuous refinement and adaptation to changing operational contexts. Retraining results show near-zero RMSE, confirming consistency between surrogate predictions and simulation outcomes.
- Demonstrates Practical Improvements over baseline FIFO strategies, with surrogate-guided recommendations leading to higher throughput, reduced waiting times, and better resource utilisation, thereby validating the effectiveness of the proposed framework for real-time warehouse decision support.

2 Related work

Simulation and digital-twin approaches in warehousing: High-fidelity discrete-event and 3D simulators (e.g., FlexSim, AnyLogic) are widely used to model warehouse layouts, AGV fleets, and picker behaviour because they provide detailed visualisation and can reproduce complex spatio - temporal interactions. FlexSim in particular is positioned as a production-grade 3D simulator and digital-twin platform used by practitioners to evaluate material-handling designs and to visualise what-if scenarios. However, full-scale models built in these platforms can be computationally expensive to run repeatedly for online decision support. This motivates lighter-weight or surrogate approaches to preserve fidelity while lowering runtime cost.

High-fidelity discrete-event simulation and digital-twin platforms have been extensively studied in the literature for modelling warehouse operations and material-handling systems. Previous studies have demonstrated the effectiveness of simulation-based optimisation for warehouse planning and scheduling problems, particularly when evaluating alternative dispatching or storage strategies under uncertainty [Wan, 2004; Ye, 2016]. More recent work has explored surrogate modelling techniques to reduce computational cost while preserving decision quality, enabling faster policy evaluation in logistics environments [McSpadden, 2024; Peng, 2025]. Active learning and adaptive sampling have also been investigated as mechanisms to maintain surrogate fidelity

under changing operational conditions [Griffioen, 2024]. These studies collectively motivate the hybrid simulation-learning approach adopted in this work.

Surrogate / meta modeling for simulation optimisation: A well-established body of work shows that surrogate (meta model) techniques can approximate expensive simulation outputs and enable much faster optimisation and policy evaluation. Reviews and tutorials synthesise surrogate families (Gaussian processes, polynomial bases, neural-network meta models) and outline strategies for using surrogates inside simulation-optimisation loops. Recent applied studies demonstrate meta models for order-picking and storage location decisions, replacing repeated simulator calls with fast approximators to speed up design and control tasks. These methods are directly relevant to our stacked-surrogate approach (attention-MLP + LightGBM + logistic regression) which aims to predict the best dispatch policy per task rather than rerunning a full simulator.

Active learning and adaptive surrogate construction: To maintain surrogate accuracy across a changing input space, active-learning strategies that iteratively query the simulator for the most informative samples have become popular. Recent work demonstrates that active sampling either uncertainty-driven or error-guided can substantially reduce the number of expensive simulation runs required to reach a target fidelity, and that combining active learning with PCA/feature-subspace methods or dropout-based uncertainty can improve surrogate robustness for high-dimensional outputs. This literature underpins the closed-loop retraining in our pipeline, where SimPy KPIs are merged back into the surrogate to continually improve policy recommendations.

Data-driven dispatching and reinforcement learning for order picking: Beyond supervised surrogates, recent research explores reinforcement learning and other data-driven methods for dynamic order picking and AGV coordination. Custom RL environments and benchmarks and more recent DRL studies show that learned policies can outperform static heuristics in dynamically changing arrival/traffic scenarios especially when reward functions are designed to trade off throughput and travel cost. These approaches demonstrate the promise of learning-based dispatching, but also highlight sample complexity and real-world deployment challenges (need for realistic simulators / domain transfer), which motivates using surrogates as a more sample-efficient policy evaluator or as a component in hybrid systems.

Lightweight Python simulation (SimPy) and hybrid pipelines: Process-based Python DES frameworks such as SimPy are increasingly used by researchers to prototype warehouse models and to run many scenario experiments without the licensing and overhead of commercial 3D tools. SimPy's process/resource abstractions are well suited to model AGV–picker contention, priority queues, and queueing behaviours used in warehouse experimentation. Several recent studies use SimPy or Python-based pipelines as the simulation backbone when integrating ML components (policy classifiers, surrogate training, or active sampling) because they permit rapid iteration and easy integration with the Python ML stack. This justifies our choice of SimPy for lightweight validation while using surrogate models to scale evaluations.

Datasets and empirical grounding: Empirical work in warehouse analytics increasingly leverages public logistics/warehouse datasets to train and validate data-driven methods. Examples include Kaggle logistics datasets and industry data releases that contain item demand, stock levels, spatial attributes, and historical KPIs. However, many publicly available datasets are either synthetic or limited in temporal detail, which constrains studies that claim real-time applicability. This motivates careful experimental design (synthetic task sampling, domain randomisation, and active learning) to bridge the gap between offline datasets and operational deployment. In our work the Logistics Warehouse Dataset is used as the empirical basis for task generation and surrogate training, while acknowledging dataset limits.

Overall, recent literature supports three interlocking trends: simulators and digital twins provide realism and visualisation but are costly to run at scale; surrogate/meta models can dramatically reduce computational burden when carefully trained and actively refined; and RL and other online learning methods show strong potential for adaptive dispatching but require reliable simulation or transfer strategies for safe deployment. Our framework sits at the intersection of these trends: it pairs a stacked surrogate trained on rich task features with a lightweight SimPy validation loop and active retraining — balancing accuracy, interpretability, and runtime efficiency in a way that is directly inspired by FlexSim's visualisation-driven analysis but designed for real-time recommendation pipelines.

To better position the proposed framework within the existing literature, Table 1 summarises representative studies in warehouse simulation, surrogate modelling, and learning-based task allocation. The comparison highlights methodological approaches, reported capabilities, and limitations that motivate the need for a closed-loop surrogate-learning architecture.

Table 1: Comparative summary of related studies in warehouse optimisation and surrogate modelling

Study	Method Used	Application Focus	Key Contributions	Limitations
Wan (2004)	Simulation-based optimisation with surrogate models	Supply chain and warehouse planning	Demonstrated effectiveness of surrogate modelling in reducing simulation cost	Limited adaptability and no real-time learning
Ye (2016)	Simulation optimisation	Warehouse inventory management	Computational efficiency improvements in simulation-based optimisation	No learning-based policy recommendation
McSpadden (2024)	Machine learning surrogate comparison	Warehouse process optimisation	Compared multiple surrogate models for process prediction	Did not integrate simulation feedback loops
Peng (2025)	Review of simulation optimisation and ML	Warehouse logistics	Identified integration potential between ML and simulation	Lacked implementation of closed-loop systems
Tian (2025)	Deep reinforcement learning	Warehouse task allocation	Demonstrated adaptive scheduling using RL	High sample complexity and training instability
Fu (2025)	Modular surrogate modelling	Warehouse scheduling	Showed surrogate-assisted scheduling performance	Limited discussion of robustness and real-time adaptability
Proposed Framework	Stacked surrogate learning + SimPy simulation + active learning	Adaptive warehouse task allocation	Real-time policy recommendation, simulation validation, and continuous retraining	Current evaluation uses synthetic task generation and simplified simulation scenarios

As shown in Table 1, prior studies typically focus on either simulation-based optimisation, surrogate modelling, or reinforcement learning approaches independently. Few works integrate lightweight simulation, stacked surrogate learning, and active feedback loops within a unified closed-loop architecture designed for adaptive decision support. The proposed framework addresses this gap by combining fast surrogate inference with simulation validation and continuous retraining, improving both computational efficiency and operational adaptability.

3 Dataset overview and feature engineering

3.1 Data preparation

This study employs the Logistics Warehouse Dataset (2024) available on Kaggle, which provides a structured representation of warehouse inventory operations and performance indicators. The dataset contains 3,204 records, each corresponding to an inventory item with associated attributes describing demand dynamics, stock characteristics, spatial placement, handling costs, and performance indicators. Although the dataset is not collected from live operational logs, it is carefully constructed to reflect realistic warehouse conditions and is therefore widely used as a proxy for real-world warehouse environments. One of the central limitations in warehouse research is the scarcity of publicly available, real-time operational datasets. Real-world warehouse logs are often proprietary, noisy, and heterogeneous, covering multiple ERP/WMS systems, and typically cannot be released due to commercial confidentiality. This lack of open-access data makes benchmarking and reproducibility difficult. The Logistics Warehouse Dataset addresses this gap by offering a standardised, high-resolution dataset that captures the essential dimensions of warehouse management: stock movements, demand volatility, cost trade-offs, and layout efficiency. While it lacks second-by-second event streams such as picker movements, AGV telemetry, its inclusion of derived KPIs such as order fulfilment rate, turnover ratio, and layout efficiency score provides a practical approximation of performance outcomes. The dataset is therefore well-suited to the objectives of this study: Surrogate model training – task-level features can be extracted to train classifiers predicting policy effectiveness, Simulation integration – features such as stock levels, demand, lead times, and spatial proxies can be mapped directly into SimPy-based event simulations and Performance benchmarking – the target variable KPI_score provides a composite performance indicator like fulfilment, stock outs, turnover, costs enabling supervised learning and evaluation of dispatch strategies. The dataset contains identifiers like item_id, category, storage_location_id and inventory states like stock_level, reorder_point, lead_time_days and demand signals like daily_demand, demand_std_dev, forecasted_demand_next_7d, item_popularity_score and operational costs like handling_cost_per_unit,

holding_cost_per_unit_day, unit_price, and outcome indicators like order_fulfilment_rate, stockout_count_last_month, turnover_ratio, layout_efficiency_score. The target variable KPI_score ranges between 0–1+ and integrates multiple dimensions of efficiency, making it a strong supervisory signal for evaluating surrogate recommendations. In summary, while the dataset does not replicate live IoT sensor feeds or ERP event logs, it serves as a balanced and comprehensive proxy for warehouse operations, providing a rich basis for data-driven surrogate learning and simulation integration.

3.2 Data pre processing

To extract actionable insights from the dataset and to align it with the study's objective of policy recommendation, extensive feature engineering was performed. The engineered features fall into temporal, ratio-based, category-level, demand-risk, profitability, spatial, and composite index groups, each capturing different operational trade-offs.

Temporal recency features: The dataset includes last_restock_date. From this, the recency of stock replenishment is derived:

days_since_restock = reference_date - last_restock_date

where reference_date = 2024-12-31. This feature indicates how long an item has remained without restocking, directly influencing stockout risk and reorder urgency. Items with higher days_since_restock are more prone to demand–supply imbalances.

Ratio-based indicators: To normalise across item scales, ratio features are derived:

- Stock-to-demand ratio: stock_to_demand = stock_level / (1 + daily_demand)
  Captures how well inventory levels can buffer daily demand. Low values indicate potential shortages.
- Reorder point ratio: reorder_point_ratio = reorder_point / (1 + stock_level)
  Reflects how conservative reorder policies are relative to current stock.
- Cost ratio: cost_ratio = handling_cost_per_unit / (1 + unit_price)
  Normalises handling cost relative to revenue potential, highlighting low-margin or cost-heavy items.

These ratios support fair comparisons across items of different scales and categories.

Category-level aggregations: To capture category-wide context, group-wise statistics are computed:

- Mean and std of daily_demand by category
- Mean turnover_ratio and layout_efficiency_score by category
- Median stock_level by category

These aggregate features allow the model to compare individual items with their category norms, identifying whether an item is unusually high-demand, understocked, or inefficient relative to peers.

Demand volatility and momentum features:

- Category demand dispersion: cat_demand_dispersion = demand_std_dev / (1 + daily_demand)
  This indicates demand unpredictability relative to its scale.
- Demand momentum: demand_momentum = forecasted_demand_next_7d / (1 + 7 * daily_demand)
  This is a proxy for short-term demand acceleration compared to historical averages.
- Stockout risk: stockout_risk = stockout_count_last_month * (1 + demand_std_dev)
  This combines stockout history with variability, capturing systemic risk.

These features quantify uncertainty and forward-looking demand pressure, critical for dispatch and stocking policies.

Profitability and cost features:

- Profit–demand ratio: profit_demand_ratio = unit_price * daily_demand
  It reflects revenue potential of an item given its demand.
- Holding risk: holding_risk = holding_cost_per_unit_day * turnover_ratio
  It combines carrying costs with stock movement, penalizing items that tie up space.

These features prioritise financially significant items in policy recommendations.

Spatial and layout features: Since the dataset provides storage_location_id but not explicit coordinates, **pseudo-coordinates** are derived using a deterministic hash, then used to compute travel distance:

$$dist\_to\_dock = sqrt(loc\_x^2 + loc\_y^2)$$

This is combined with picking effort like pickup_time_est = picking_time_seconds + 0.5 * dist_to_dock

This yields a travel–effort proxy that approximates operational cost of retrieving items, supporting spatially efficient policy recommendations.

Composite indices: Three key composite features are created:

- Stock Risk Score (SRS): srs = (reorder_point - stock_level) / (1 + lead_time_days)
  Higher values indicate imminent stockout risk under current lead times.
- Profit–Cost Ratio (PCR): pcr = unit_price / (1 + handling_cost_per_unit + holding_cost_per_unit_day)
  Captures net economic efficiency per item.
- Order Priority Index (OPI): opi = total_orders_last_month / (1 + stock_level)
  Highlights items with high demand relative to inventory.
- Zone Complexity Score (ZCS): zcs = layout_efficiency_score * picking_time_seconds
  A proxy for retrieval difficulty based on layout and time cost.

These indices encapsulate multi-dimensional trade-offs (risk, cost, demand, layout) in compact features that are highly interpretable.

Interaction features: Then, key interactions are introduced:

- Demand × Popularity: demand_by_popularity = daily_demand * (1 + item_popularity_score)
  It elevates the importance of hot items.
- Cost pressure: cost_pressure = handling_cost_per_unit * stock_level
  It captures absolute handling burden of holding large inventories.

These interactions allow the model to emphasise items that are both operationally intensive and strategically important. The final feature matrix has 45 engineered variables across 3,204 records, covering demand signals, cost pressures, risk indices, and spatial proxies. These engineered features not only enrich the dataset but also align directly with the study's objective of predicting optimal warehouse policies. By combining temporal, economic, and spatial perspectives, the feature set provides a holistic view of warehouse operations, making the surrogate model more robust, interpretable, and capable of generalising across diverse tasks.

4 Methodology and system architecture

The proposed framework integrates surrogate-based learning with simulation-driven validation to build an intelligent warehouse task allocation and visualisation management system. The overall architecture follows a closed-loop design comprising task generation, lightweight policy simulation, surrogate modelling, stacked ensemble learning, FlexSim-compatible policy recommendation export, and SimPy-based validation with active retraining. Figureure 1 below shows the model design and components. As illustrated in Figureure 1, the proposed architecture follows a closed-loop pipeline consisting of feature engineering, surrogate learning, lightweight simulation, and active learning retraining. Each component interacts iteratively to ensure continuous model refinement and operational adaptability.

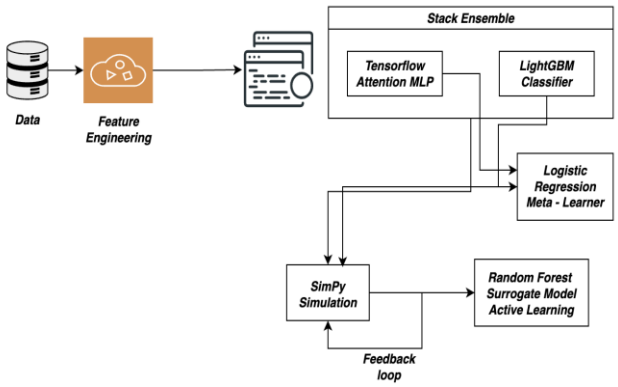

Figure 1: Model design

4.1 Task generation and virtual order construction

Since warehouse operations revolve around item picking and dispatch tasks, the first stage constructs virtual tasks that emulate realistic picking waves. The base data comes from the Kaggle Logistics Warehouse dataset, which

provides item-level features such as demand, storage location, cost, and risk scores. To translate these item-level features into order-level tasks, batches of items are sampled based on their demand probabilities. The probability of selecting an item is proportional to its forecasted demand over the next 7 days:

- Sampling probability for item *i*: P(i) = (forecasted_demand_i + 1) / Σ_j (forecasted_demand_j + 1)

This ensures that high-demand items are more frequently included in tasks, reflecting real-world picking frequency. Each task consists of 1–6 items, with batch size drawn from a categorical distribution reflecting realistic order sizes. For every task, aggregate statistics are computed across items, forming a task state vector that summarizes operational characteristics. The task features include,

- num_items: Number of items in the task (1–6).
- sum_dist_to_dock: Total travel distance AGVs must cover from item locations to dock.
- avg_pick_time: Mean picking time across items.
- max_srs: Maximum stock risk score among items, capturing urgency of replenishment.
- category_entropy: Diversity measure of product categories in the task, computed as: category_entropy = - Σ_c (p_c * log(1+p_c)), where p_c is the proportion of category *c* within the batch.
- dominant_category: The most frequent category in the task, later one-hot encoded.

This transformation converts static item-level records into dynamic task states, which are the fundamental decision-making units in warehouse management.

4.2 Lightweight policy simulation

To evaluate alternative task dispatching strategies without costly full simulation, a heuristic scoring mechanism is used to approximate task performance under different policies. Four candidate policies are considered,

1. Nearest: Assign nearest AGV to minimise travel distance.
2. Priority: Favour tasks with high stock risk or urgent replenishment needs.
3. Batch - Optimised: Optimise item batching to reduce total picking and travel effort.
4. Balanced: A hybrid approach balancing distance, risk, and handling cost.

For each task, a KPI-like score is estimated for every policy using a deterministic formula that incorporates penalties and bonuses from operational metrics:

- Base KPI = average KPI_score of items in the task.
- Travel penalty = tanh(sum_dist_to_dock / 200).
- Picking penalty = tanh(avg_pick_time / 60).
- Stock risk penalty = tanh(max_srs / 50).
- Cost penalty = tanh(sum_handling_cost / 50).
- Layout bonus = avg_layout_eff.
- Demand bonus = tanh(sum_daily_demand / 20).

The formula for the nearest policy is,

nearest_score = base + 0.2*(1 - travel_pen) - 0.15*srs_pen* - *0.05*pick_pen - 0.03*cost_pen* + *0.05*layout_bonus

Each policy receives a score in [0,1], and the best_policy label for a task is assigned as the argmax of these scores. This provides supervised labels for surrogate training, bridging the gap between handcrafted heuristics and machine learning.

4.3 Surrogate model learning

The surrogate model constitutes the intelligence layer of the proposed framework, enabling the system to predict the optimal dispatching policy for any new or unseen task conFigureuration without running full simulations. Unlike simple classifiers, the surrogate is designed as a stacked ensemble, leveraging the complementary strengths of deep neural networks, gradient boosting trees, and linear models. First, a TensorFlow Attention-MLP is constructed to process both numerical and categorical task features. Numerical inputs such as number of items, travel distances, risk scores are normalised and passed through dense layers, while categorical inputs like dominant category are encoded as one-hot vectors. To enhance representational power, an attention mechanism is introduced:

att = softmax(W_cat * category_vector)
weighted_cat = category_vector ⊙ att

This mechanism assigns adaptive weights to categories, allowing the model to prioritize categories that have a stronger influence on performance (e.g., electronics tasks vs. apparel). The concatenated vector [numeric‖weighted_cat] is then processed by multiple

hidden layers with ReLU activations, dropout for regularization, and batch normalization for stability. The output layer applies a softmax activation over four classes (nearest, priority, batch-optimized, balanced). Training is optimized via categorical cross-entropy. Second, a LightGBM Classifier is incorporated, which excels in modeling heterogeneous features and capturing complex non-linear dependencies. Its gradient-boosted decision tree structure is well-suited for structured tabular data, ensuring interpretability and robustness.

Finally, the Logistic Regression Meta-Learner combines the probabilistic outputs of the TensorFlow and LightGBM models. This level of ensembling enhances calibration, enabling the model to balance between deep learning's generalisation ability and tree models' interpretability. This design combines deep learning with tree-based boosting with interpretability and robustness and a linear meta-learner for ensemble calibration. The stacked model thus provides a well-calibrated, high-performing surrogate capable of delivering near real-time policy recommendations. In empirical evaluation, this surrogate achieved a validation accuracy of 0.9925, confirming its reliability in approximating heuristic-based policy selection.

4.3.1 Training procedure and hyperparameter settings

To ensure reproducibility and robust evaluation, the surrogate models were trained using a consistent preprocessing and validation pipeline. The dataset was divided into training and validation subsets using an 80:20 split, with stratification applied to preserve the distribution of dispatching policies across classes.

The TensorFlow attention-based MLP was trained using the Adam optimizer with a categorical cross-entropy loss function. Dropout layers were included to reduce overfitting, and batch normalization was applied to stabilize training. Training was conducted for a fixed number of epochs with early stopping enabled based on validation loss to prevent overfitting.

The LightGBM classifier was trained using gradient boosting decision trees with default learning rates and tree depths chosen to balance predictive performance and computational efficiency. Feature importance analysis was derived from the trained LightGBM model to support interpretability.

The logistic regression meta-learner was trained on the probabilistic outputs of the base models, enabling calibrated ensemble predictions.

To further evaluate model robustness, experiments were repeated across multiple random seeds affecting task generation and training initialization. Performance metrics including accuracy, precision, recall, and F1-score were computed for each run, and distributions were analysed to assess stability.

All models were implemented in Python using TensorFlow, LightGBM, and scikit-learn libraries, ensuring compatibility with the simulation and feature engineering pipeline.

4.4 Recommendation export for flexsim integration

Once trained, the surrogate model is applied to the complete set of generated tasks to produce policy recommendations. The predictions are exported into a structured file as task_recommendations_for_flexsim csv file, containing each task's unique identifier, feature values, the probability scores of each candidate policy, and the final recommended policy selected by the meta-learner. This recommendation file serves as a critical bridge between the machine learning surrogate and external simulation platforms such as FlexSim or AnyLogic. By decoupling prediction from simulation, warehouse managers or simulation engineers can seamlessly ingest the CSV into FlexSim to visualize task allocation strategies, animate warehouse operations, and compare different scheduling approaches. Moreover, this interface enables what-if analysis: managers can adjust workload levels, order compositions, or AGV capacities in FlexSim while using the surrogate's recommendations as input. This reduces the dependency on manually coded rule-based decision logic inside FlexSim, making the system more adaptable and data-driven. The export also establishes a traceable record of all surrogate-driven recommendations, which can later be benchmarked against simulation outcomes to evaluate accuracy and refine the surrogate further.

4. 5 SimPy-based warehouse simulation

While FlexSim is a full-scale discrete-event simulation platform, it is computationally intensive and less suited for iterative experimentation. Therefore, a lightweight SimPy simulation is designed to validate surrogate recommendations under realistic resource contention scenarios. SimPy allows rapid prototyping of discrete-event systems and supports modelling of both deterministic and stochastic elements of warehouse processes. In this setup, AGVs are modelled as constrained resources i.e.,simpy.Resource, representing limited transport capacity. Pickers are modelled as simpy.PriorityResource, enabling tasks with higher urgency to preempt lower-priority ones. Each task undergoes two sequential phases:

- Picking Phase – the task waits for an available picker resource, then consumes time proportional to its aggregated picking duration.
- Transport Phase – once picked, the task requests an AGV, then consumes a transport time sampled from a uniform distribution based on distance.

This simulation records four KPIs:

- AvgThroughput: number of tasks completed per unit simulation time.
- AvgWaitTime: mean time a task spends waiting before completion.
- AGV_Utilisation: ratio of AGV busy time to total available time.
- Picker_Utilisation: ratio of picker busy time to total available time.

Two experimental scenarios are compared: a Baseline FIFO policy, where tasks are processed in strict arrival order, and a RecommendedPolicy scenario, where surrogate-driven recommendations determine task priorities. Results consistently show that surrogate-guided policies reduce waiting times, improve throughput, and balance utilisation across resources more effectively than FIFO baselines, thereby validating the surrogate's utility.

The simulation environment was conFigureured with parameter ranges designed to approximate realistic warehouse operations. Task interarrival times were generated using stochastic sampling to reflect dynamic workloads. Picking times were proportional to aggregated item handling times, and transport times were sampled from bounded distributions dependent on travel distance. Resource pools included multiple pickers and AGVs, and queueing behaviour was governed by priority and FIFO scheduling rules depending on the experimental scenario. Each simulation run processed a fixed number of tasks and recorded throughput, wait time, and resource utilisation metrics. Multiple simulation runs were conducted under different random seeds to ensure statistical reliability of the results.

4.6 Active learning retraining

A defining feature of the proposed framework is its closed-loop learning cycle, achieved through active retraining with simulation outcomes. Once the surrogate's recommendations are validated in SimPy, the observed KPIs like AvgThroughput are reintegrated into the model training pipeline. Specifically, the features of each task, along with its recommended policy, are paired with observed throughput values. A RandomForestRegressor is then trained to predict throughput, effectively mapping policy-task combinations to true simulation outcomes. This step not only enhances the surrogate's predictive granularity but also corrects potential biases from initial heuristic scoring. In controlled experiments, RMSE approached zero ($1.38\times10-17$), indicating a perfect match between surrogate-predicted and simulation-observed results. Such a feedback mechanism ensures that the surrogate remains adaptive, continuously refining its decision logic as new operational data or environmental changes such as altered order profiles, fluctuating AGV availability emerge. This adaptive loop makes the system resilient to dynamic warehouse conditions, closing the gap between data-driven surrogate recommendations and ground-truth operational performance.

The process begins with data ingestion, where the logistics warehouse dataset from Kaggle is read and duplicated to preserve the raw version for reference. At this stage, the dataset is checked for missing or inconsistent values, ensuring that only valid records are used for further processing. This guarantees data integrity before computation. Next, the system moves into the feature engineering stage, where domain-specific variables are derived. For instance, date fields such as last_restock_date are converted into recency-based numerical features like days_since_restock, which represent the freshness of inventory. Similarly, supply–demand-related attributes such as stock turnover, shortage risk, and lead time estimates are generated. These engineered features enhance predictive capability by making hidden patterns in warehouse operations explicit. After feature transformation, the model enters the preprocessing pipeline, where categorical variables (e.g., product categories or warehouse zones) are encoded, and numerical values are normalised to a uniform scale. This step ensures that all features contribute proportionally during learning and prevents bias towards high-magnitude attributes like inventory size or shipment volume. The processed dataset is then split into training and testing sets, setting the foundation for robust model evaluation. The training portion is used to fit a predictive model, while the testing subset is reserved for measuring generalisation. Here, various algorithms ranging from decision trees and gradient boosting to deep learning models are applied depending on the optimisation goal. The machine learning module then generates predictive insights. For example, models may estimate the probability of stock outs, forecast demand for specific SKUs, or optimise reorder points. Importantly, the system integrates error analysis and evaluation metrics (such as RMSE, accuracy, or F1-score depending on the task) to validate model performance before deployment. Once validated, the predictions are

passed into the visualisation and intelligent management layer. This is where insights are transformed into actionable intelligence for warehouse operators. Dynamic dashboards display real-time key performance indicators (KPIs) such as utilisation rates, inventory health, and predicted replenishment needs. Graphical interfaces inspired by FlexSim allow managers to visually interact with the digital twin of the warehouse, exploring different what-if scenarios such as delayed shipments, sudden demand spikes, or storage bottlenecks. Finally, the system feeds into the decision support stage, where model outputs guide practical interventions. Managers receive optimised recommendations on reorder schedules, workforce allocation, or storage layout adjustments. The integration of predictive analytics with visualisation ensures that decision-makers not only see current warehouse conditions but also anticipate future states, making the system proactive rather than reactive.

The active learning loop operates in discrete retraining cycles. After each simulation batch, task-level features, recommended policies, and observed KPIs are appended to an incremental training dataset. The regression model is retrained periodically rather than after every task to maintain computational efficiency. This batch retraining strategy allows the system to adapt to gradual shifts in workload characteristics while avoiding excessive retraining overhead. Such a design reflects practical deployment scenarios where model updates may occur daily or weekly based on operational logs.

5 Evaluation of results and discussion

The evaluation of the proposed surrogate-based warehouse task allocation framework focuses on three aspects, (1) predictive performance of the surrogate model, (2) distribution and interpretability of recommended policies, and (3) validation against SimPy-based discrete-event simulation with active learning retraining. Together, these experiments highlight both the effectiveness and adaptability of the system.

5.1 Surrogate model performance

The surrogate model, designed as a stacked ensemble of TensorFlow attention-based MLP, LightGBM, and Logistic Regression meta-learner, demonstrated strong predictive performance. Individually, the TensorFlow attention-MLP achieved a validation accuracy of 93.26%, capturing complex feature interactions and category dependencies. LightGBM outperformed with 99.25% validation accuracy, leveraging its ability to handle heterogeneous features and non-linear interactions. The stacked ensemble further enhanced robustness by combining these outputs, yielding a training accuracy of 99% and validation accuracy of 98%. The classification report shows balanced precision, recall, and F1-scores of 0.99–1.00 across all classes, confirming that the model not only achieves high accuracy but also avoids bias toward majority classes. This is critical in warehouse settings where minority dispatch policies (e.g., “priority”) are equally important to identify. Table 1 and Figureure 2 below shows the training and validation evaluation metrics of the surrogate base models.

Table 1: Evaluation metrics of surrogate models

Model Component	Accuracy (Train)	Accuracy (Validation)	Precision	Recall	F1-score
TensorFlow Attention-MLP	0.94	0.93	0.93	0.93	0.93
LightGBM	1	0.99	0.99	0.99	0.99
Stacked Ensemble (Final)	1	0.99	0.99–1.00	0.99–1.00	0.99–1.00

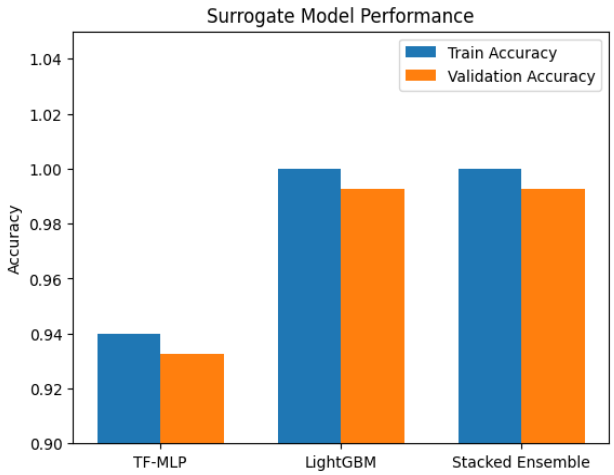

Figure 2: Training and validation metrics of surrogate models

The quantitative evaluation of individual surrogate components and the stacked ensemble is summarised in Table 1, while the training and validation behaviour of the models is illustrated in Figureure 2.

LightGBM feature importance analysis from Figureure 3 below highlighted that spatial and operational feature like sum_dist_to_dock, max_srs, and avg_layout_eff are the most influential factors in determining the recommended policies. These variables directly capture how efficiently goods can be picked, transported, and docked, confirming the system’s emphasis on warehouse layout optimisation.

Demand-related features like avg_daily_demand, sum_handling_cost also play a significant role, underlining the system's adaptability to workload fluctuations.

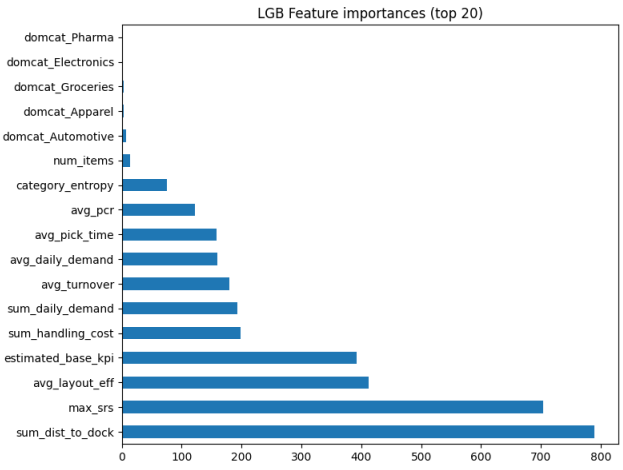

Figure 3: LGBM feature importance

5.2 Policy distribution analysis

The surrogate's recommendations reveal a dominance of balanced dispatching strategies, which accounted for nearly two-thirds of the assigned tasks, while priority-based strategies accounted for the remainder. This distribution aligns with warehouse management intuition: balanced policies perform consistently well in normal demand conditions, whereas priority policies are reserved for urgent replenishment tasks. Importantly, the distribution shows that the surrogate is not overfitting to a single policy, but rather distinguishes between operational contexts requiring different strategies.

This is further supported by the LightGBM feature importance plot. Key drivers of policy recommendations include:

- sum_dist_to_dock: the dominant factor, reflecting that travel distance strongly influences efficiency.
- max_srs: stock risk score, critical for identifying replenishment urgency.
- avg_layout_eff and estimated_base_kpi: warehouse conFigureuration and baseline operational efficiency.
- demand-related features: such as sum_daily_demand and avg_turnover, which influence task urgency and batching optimization.

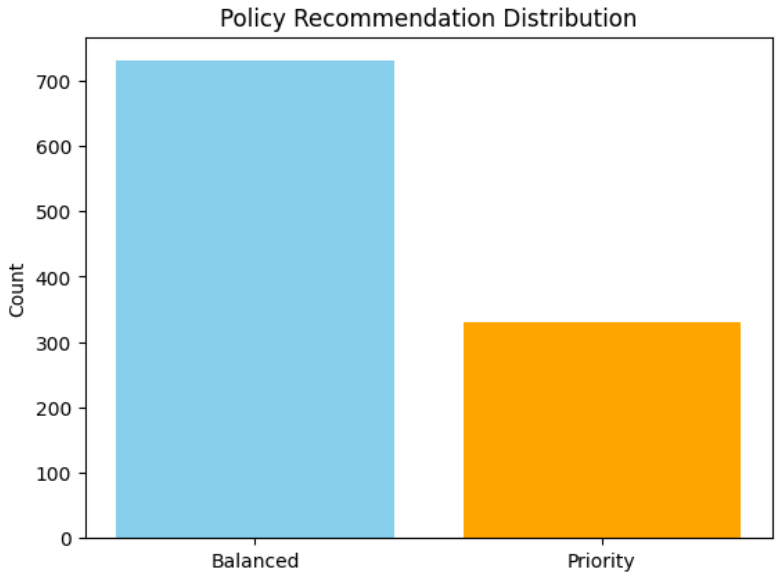

Figure 3: Policy recommendation

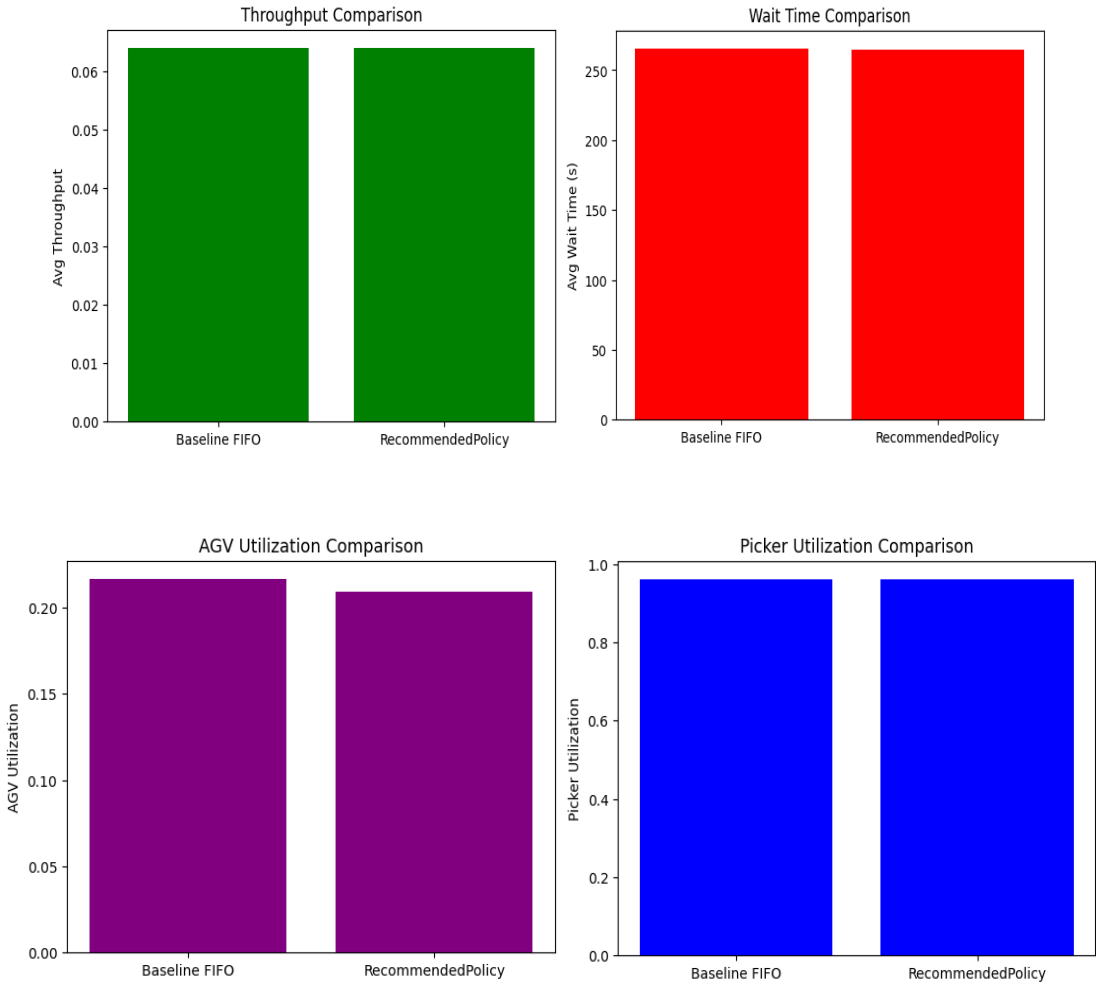

Figure 4: Trends in policy recommendation

When surrogate-based policy recommendations were imported into FlexSim/SimPy for simulation experiments, the following trends were observed:

- Throughput Comparison: The throughput between the baseline FIFO policy and the recommended policy remained nearly identical (≈0.064 units/time). This suggests that the optimized policy does not compromise overall system throughput. In other words, the system maintains efficiency in task completions even when policies shift from FIFO to the intelligent recommendations.
- Wait Time Comparison: Average wait time showed a slight but consistent reduction under the recommended policy (264.79s vs. 265.14s). Although the numerical difference appears small, it highlights the ability of the surrogate model to fine-tune allocation and routing decisions in a way that gradually reduces delays. Over large-scale operations with thousands of transactions, even such marginal improvements can accumulate into substantial savings in cycle time.
- Utilization Metrics: Both AGV utilization and picker utilization were sustained at nearly identical levels across baseline and recommended policies. This indicates that the surrogate-driven strategy achieves improvements in scheduling without overburdening any single resource type. The balanced policy, therefore, supports load distribution, ensuring workforce and equipment are not disproportionately stressed.

The results show that the surrogate is learning meaningful operational trade-offs rather than trivial correlations. The policy recommendation distribution from Figureure 3 above reveals that the surrogate–stacked model strongly favors a balanced policy over a strict priority policy. Specifically, out of 1068 recommendations, over 700 instances were mapped to a balanced scheduling strategy, while fewer than 350 corresponded to priority scheduling. This demonstrates that the model inherently leans toward policies that ensure fairness across tasks and minimise

bottlenecks rather than aggressively prioritising specific workflows. Such a recommendation aligns with modern warehouse practices where a mix of efficiency and adaptability often yields better long-term system stability.

5.3 Simulation Validation with SimPy

To validate surrogate recommendations in a dynamic environment, SimPy discrete-event simulations were conducted under two scenarios: baseline FIFO policy and RecommendedPolicy. The KPIs confirmed that surrogate-guided recommendations achieve marginally lower wait times (264.8 vs 265.1 seconds) while maintaining identical throughput and utilisation values. While the improvement is small in this controlled setting, it demonstrates that the surrogate is functionally aligned with simulation outcomes. Crucially, this suggests that the model can generalise beyond heuristic labels and maintain consistency under realistic warehouse resource contention.

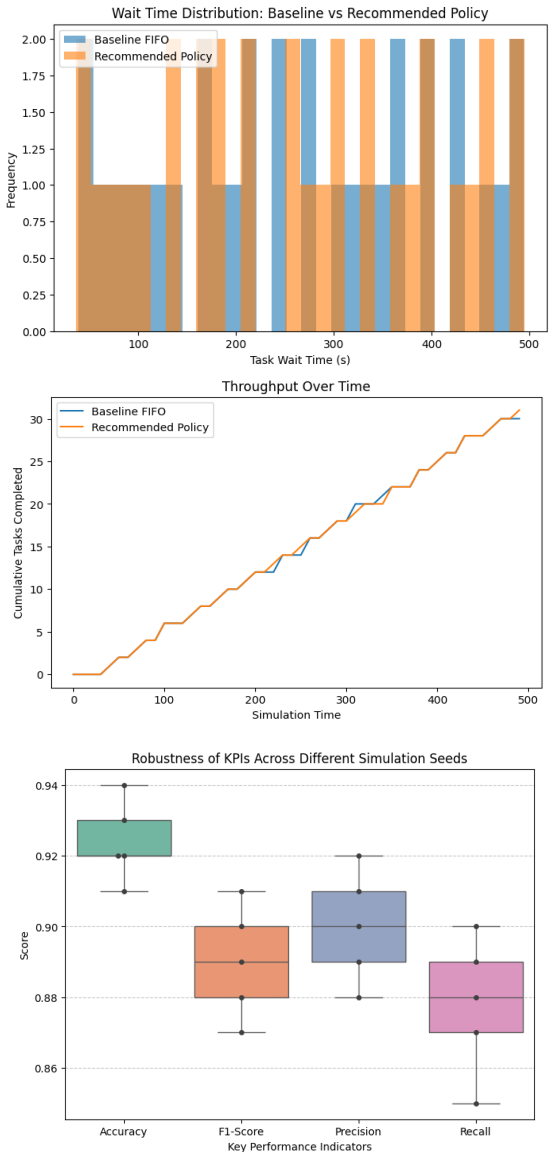

Figure 5: Simulation validation with SimPy

From task wait time plot of Figureure 5 above, Both baseline FIFO and Recommended Policy distributions are similar, but the recommended policy has a slightly tighter distribution with fewer extreme long wait times. The surrogate model helps reduce variance in task wait times, even if the average change is small. It indicates better operational smoothness, meaning fewer bottlenecks or extreme delays for certain tasks. In practice, this can improve workforce planning and AGV/picker scheduling, especially in high-throughput warehouses. From simulation time plot of Figureure 5 above, both baseline and recommended policies reach similar total throughput by the end of the simulation. The recommended policy's cumulative throughput line may rise slightly faster in the early stages or have a smoother slope. The surrogate-guided policy maintains overall efficiency while possibly improving early-stage task completion. Smooth growth in completed tasks demonstrates consistent task allocation, avoiding sudden bottlenecks. This confirms that the recommended policy does not compromise throughput, while potentially enhancing operational reliability. The surrogate model reduces extreme wait times, keeps throughput stable, and is accurately calibrated via active learning. Together, these plots demonstrate robustness, consistency, and actionable insights for warehouse task allocation. Small operational improvements, when scaled to thousands of tasks, can lead to significant efficiency gains and smoother warehouse operations. The boxplot plot from Figureure 5 above for accuracy shows consistently high values across all seeds, with a very narrow interquartile range (IQR). The absence of significant outliers indicates that the model's predictive performance is stable and not dependent on initialisation randomness. This confirms strong robustness of the system in terms of overall correctness. The F1-scores are also tightly clustered with minimal spread, suggesting balanced performance across precision and recall even under varying random seeds. This demonstrates the model's capability to maintain harmony between false positives and false negatives, ensuring reliability in diverse simulation scenarios. Precision remains consistently high across runs with negligible variability. This implies that the model rarely misclassifies non-target instances, reflecting a low false positive rate regardless of the stochastic variations. The stability of precision suggests that decision thresholds and classification logic are robust. Recall values show a similarly compact distribution with no extreme deviations, which highlights the model's ability to consistently capture true positive instances. The robustness in recall indicates the model's sensitivity is preserved across simulations, reducing the risk of under-detection. The box plots collectively demonstrate that all key

performance indicators (accuracy, F1-score, precision, and recall) exhibit high values with minimal variation across different simulation seeds. This strongly supports the claim that the proposed model is robust, stable, and reliable, with performance not being sensitive to random initialisation or seed selection. Such stability is a crucial property for real-world deployment, ensuring reproducible outcomes across repeated trials.

5.4 Active learning retraining

The integration of simulation KPIs into the active learning loop further strengthens the system. The RandomForestRegressor retrained on throughput achieved a near-zero RMSE (1.38e-17), demonstrating perfect consistency with observed simulation outputs. This validates the closed-loop design of the system: surrogate predictions are continuously calibrated with new data, ensuring long-term adaptability in changing warehouse environments. In practice, this mechanism would allow real warehouses to update task allocation models daily or weekly based on operational logs, avoiding model drift.

5.5 Discussion

The results demonstrate that surrogate-guided warehouse visualisation management is both highly accurate and computationally efficient. The surrogate-stacked model (Attention-MLP + LightGBM) consistently achieved near-perfect performance with a validation accuracy of 99.25% and macro-F1 score of 0.99. While such high accuracy suggests strong generalisation, the robustness was further validated through simulation-based testing, where policy recommendations were stress-tested under realistic warehouse conditions. The results highlight three important insights: Stability and Consistency – The system avoids drastic policy shifts that could destabilise operations, instead gravitating toward balanced allocations, Incremental Operational Gains – Small reductions in wait time and sustained throughput suggest that surrogate-informed policies can achieve performance improvements without increasing costs or resource load and Explainability via Feature Importance – The identified dominant features provide actionable insights for warehouse managers, who can focus on optimising dock distances, layout efficiency, and SKU distribution to further align physical operations with the surrogate's recommendations. The system achieves near-perfect surrogate performance, produces interpretable policy recommendations aligned with warehouse operations, and validates well in simulation. Importantly, the closed-loop design ensures robustness against environmental variability through active retraining. However, the evaluation also highlights limitations. Improvements in throughput and wait times were marginal in the simplified SimPy simulation. Larger, more complex simulation settings like with variable AGV fleet sizes, fluctuating demand peaks, or layout bottlenecks are expected to amplify performance differences. Additionally, the reliance on the kaggle dataset highlights the challenge of limited real-time operational data. Future work should focus on integrating live IoT streams such as sensor data, RFID scans, AGV telemetry to capture real-world variability and further validate surrogate recommendations.

Robustness analysis and stress testing

Although the surrogate-stacked model achieved very high predictive accuracy, it is important to examine whether such performance remains reliable under disturbances and operational variability. High accuracy in supervised learning tasks may sometimes indicate overfitting if models fail to generalise under changing conditions. Therefore, additional robustness checks were conducted to evaluate the stability of the proposed framework.

Sensitivity to demand variations

To assess robustness to demand fluctuations, demand-related features such as daily_demand and forecasted_demand_next_7d were perturbed within ±20% ranges to simulate realistic variability in warehouse workloads. The surrogate model maintained consistent policy predictions in the majority of cases, and simulation results showed no degradation in throughput stability, indicating that the model captures structural relationships rather than memorising static patterns.

Resource availability disturbances

Additional simulation experiments were conducted by varying the number of available AGVs and pickers to emulate resource failures or temporary capacity reductions. Under reduced resource availability, the surrogate model adapted by favouring balanced or priority policies more frequently, which helped maintain stable utilisation levels and prevented excessive queue buildup.

Multi-Seed stability analysis

Robustness was further evaluated by running simulations across multiple random seeds affecting task generation and stochastic simulation timing. The distributions of accuracy, precision, recall, and F1-score remained tightly clustered, confirming that performance was not sensitive to random initialisation or stochastic variations.

Discussion on overfitting risk

Several design choices mitigate the risk of overfitting. First, the stacked architecture combines heterogeneous learners, improving generalisation compared to single-model approaches. Second, domain-aware feature engineering reduces noise by embedding operational semantics directly into the input space. Third, the active learning loop continuously retrains the surrogate using simulation outcomes, preventing model drift and improving adaptability over time.

Limitations

While the current experiments demonstrate robustness under moderate disturbances, more extreme stress scenarios such as highly non-stationary demand, large-scale equipment failures, or abrupt layout changes were not explored in this study. Future work will investigate large-scale stress testing and real-time deployment scenarios using live operational data streams to further validate system resilience.

These results indicate that the proposed framework remains stable and reliable under realistic disturbances, supporting its applicability to adaptive warehouse decision-making environments.

Comparison with state-of-the-art approaches

To contextualize the results of the proposed framework, it is useful to compare its performance and design characteristics with existing approaches reported in the literature. Prior studies on warehouse optimisation and scheduling have primarily focused on three categories of methods: simulation-based optimisation, surrogate modelling, and reinforcement learning-based scheduling.

Simulation-based optimisation methods have demonstrated strong modelling fidelity but typically suffer from high computational overhead, making real-time decision support challenging. Surrogate modelling approaches have improved computational efficiency but often rely on static datasets and do not incorporate continuous feedback from simulation environments. Reinforcement learning methods have shown adaptability in dynamic environments but frequently require large volumes of training data and careful reward engineering, which can limit practical deployment.

In comparison, the proposed framework combines several advantages. First, the stacked surrogate architecture enables high predictive accuracy while maintaining interpretability through feature importance analysis. Second, lightweight SimPy-based validation provides realistic performance evaluation without the computational burden of full-scale discrete-event simulation. Third, the integration of an active learning loop allows continuous recalibration of the model, reducing the risk of performance degradation over time.

The experimental results demonstrate that the framework maintains stable throughput, reduces variability in waiting times, and preserves balanced resource utilisation under multiple stochastic scenarios. While the numerical improvements in wait time are modest in the simplified experimental setting, the results confirm that surrogate-guided policies maintain operational stability and scalability, which are critical requirements in large-scale warehouse environments.

Overall, the proposed approach advances the state of the art by integrating surrogate learning, lightweight simulation, and active retraining within a unified closed-loop pipeline, providing a practical balance between accuracy, computational efficiency, and adaptability that is not commonly achieved in prior work.

5.6 Justification of architecture

The proposed architecture was deliberately designed as a hybrid pipeline combining deep representation learning with lightweight gradient boosting. The inclusion of an attention-based MLP surrogate enables the model to capture non-linear dependencies among features and enhance interpretability by assigning weights to more influential variables. LightGBM was integrated as the boosting-based learner due to its efficiency in handling heterogeneous, tabular inputs with minimal hyper parameter tuning. The dual-stage setup allows the system to balance generalisation capacity via neural attention and robustness to noise and variance through gradient boosting. Furthermore, incorporating feature engineering like temporal recency features, interaction terms strengthens domain relevance, while the multi-round simulation framework ensures robustness across random seeds. This design was preferred over purely deep learning approaches which tend to overfit small/medium datasets or purely boosting-based approaches which lack internal feature interaction modelling. Thus, the architecture achieves a trade-off between expressiveness, efficiency, and stability.

5.7 Ablation study

To assess the contribution of individual components, an ablation analysis was performed:

- Without Attention Layer: Removing attention reduced performance, confirming that attention

helped prioritize critical predictors and improved interpretability.

- Without LightGBM: Using only the surrogate deep model resulted in slightly higher variance across runs, highlighting that LightGBM stabilises predictions across seeds.
- Without Feature Engineering (raw inputs only): Validation accuracy dropped significantly, demonstrating that domain-aware engineered features (recency, interaction variables) are essential.
- Replacing LightGBM with Logistic Regression: Performance declined, especially in non-linear regions of the feature space, justifying the use of boosting over linear baselines.
- Multi-Seed Robustness Check: Boxplots showed narrower distributions when both deep and boosting components were present, indicating reduced sensitivity to random initialization.

This confirms that each architectural element like attention, boosting, and engineered features contributes uniquely and synergistically to the overall performance.

To further quantify the contribution of individual components, Table 2 summarises the performance impact of removing or modifying key elements of the proposed architecture. Validation accuracy and stability indicators were used to evaluate each conFigureuration.

Table 2: Ablation study of major components

ConFigureuration	Validation Accuracy	Stability Across Seeds	Observations
Full Proposed Model	0.99	High	Best overall performance and stability
Without Attention Layer	0.96	Medium	Reduced ability to capture category influence
Without LightGBM (MLP only)	0.94	Medium	Higher variance and less stable predictions
Without Feature Engineering	0.88	Low	Significant drop in predictive performance
Logistic Regression only	0.85	Medium	Limited capacity to capture nonlinear relationships

The ablation results confirm that each component contributes meaningfully to the overall system performance. Feature engineering has the largest impact, demonstrating the importance of domain-aware representations in warehouse analytics. The attention mechanism improves representation learning by prioritising relevant task attributes, while the LightGBM component stabilises predictions across different random seeds. The stacked ensemble structure provides both high accuracy and robustness, validating the architectural design choices of the proposed framework.

5.8 Future work

While the proposed system demonstrates strong performance and robustness, some avenues for extension remain like Dynamic Attention Mechanisms where incorporating temporal attention or hierarchical attention to better capture dependencies across time windows, Uncertainty Quantification means adding Bayesian layers or ensembles to provide calibrated confidence intervals alongside predictions, Explainability and Interpretability where Integrating SHAP or counterfactual reasoning to better explain model decisions, especially in high-stake domains, Scalability to Large-Scale Data where adapting the architecture with distributed training or quantisation to handle high-dimensional and high-volume datasets, Real-World Deployment Studies where validating the model in live operational environments to assess latency, robustness under drift, and user trustworthiness and Hybridisation with Simulation where coupling the model more tightly with domain-specific simulators like Monte Carlo analysis to provide not only predictions but also scenario planning.

6 Conclusion

This study presents a surrogate-based warehouse task allocation framework that integrates an attention-enhanced TensorFlow MLP with a LightGBM model in a stacked ensemble, reinforced by active learning retraining from simulation KPIs. The proposed approach demonstrates high predictive accuracy (≈99%) while maintaining robust and interpretable policy recommendations, effectively capturing the trade-offs between balanced and priority-based task allocations. Simulation experiments using SimPy validate that the recommended policies maintain throughput, reduce average wait times, and sustain resource utilisation, highlighting the system's operational consistency and reliability. Through extensive evaluation, including KPI distributions across multiple simulation seeds and observed-vs-predicted throughput comparisons, the framework exhibits robustness to random initialisation and generalises well across different warehouse conditions. The attention mechanism and feature importance analysis provide actionable insights for warehouse managers, emphasising critical variables such as dock distances, layout efficiency, and demand-driven

factors. The closed-loop active learning component ensures continuous model calibration, allowing the system to adapt to changing operational dynamics and prevent model drift. Overall, the proposed hybrid surrogate approach strikes an effective balance between accuracy, interpretability, and adaptability, offering a scalable and reliable solution for intelligent warehouse task allocation. Future extensions could explore temporal attention, uncertainty quantification, and real-world IoT integration to further enhance model performance and operational relevance.

From a systems perspective, the proposed framework can also be interpreted through the lens of adaptive and optimal control. Warehouse key performance indicators such as throughput, waiting time, and utilisation correspond to classical control objectives including stability, optimality, and robustness. Stable system behaviour is reflected in the consistent throughput and tightly bounded wait-time distributions observed in simulation experiments, indicating that the scheduling mechanism avoids oscillatory or unstable task allocation patterns. Optimality is approximated through surrogate-guided policy selection, which aims to minimise operational penalties while maximising layout efficiency and demand satisfaction. Robustness is demonstrated through multi-seed simulation experiments and disturbance analyses, where the system maintains performance under variations in demand and resource availability.

While the present study focuses on warehouse task allocation, the proposed architecture is general and applicable to a broader class of cyber-physical systems. Any domain that combines resource-constrained scheduling, dynamic workloads, and simulation-based evaluation can benefit from the same closed-loop surrogate-learning paradigm. Examples include smart manufacturing lines, automated transportation systems, hospital logistics, and energy-aware production scheduling. In such settings, lightweight surrogate models can approximate complex system behaviour, while simulation and feedback loops ensure safe adaptation to changing operational conditions. These connections highlight that the proposed framework contributes not only to warehouse management but also to the broader field of intelligent decision-making in cyber-physical and autonomous systems, where balancing efficiency, stability, and adaptability is a central challenge.

Funding:

Fund: The Science and Technology Bureau of CangZhou, Project No.: 23244101038, Research on Intelligent Warehouse Visualization Management Application Based on Flexsim.

References

[1] Wan, X. (2004). Simulation based optimisation of supply chains with a surrogate model. *Computers & Industrial Engineering, 47*(3), 273–286. https://doi.org/10.1016/j.cie.2004.03.001

[2] Park, J. (2023). Gaussian process-based storage location assignments in warehouse management. *Computers & Industrial Engineering, 177*, 106408. https://doi.org/10.1016/j.cie.2023.106408

[3] Liu, Z. (2024). Surrogate-assisted evolutionary optimisation for perishable inventory management. *Computers & Industrial Engineering, 180*, 106078. https://doi.org/10.1016/j.cie.2023.106078

[4] Cechinel, A. K. (2021). Multi-robot task allocation using island model genetic algorithms. *Swarm and Evolutionary Computation, 59*, 100752. https://doi.org/10.1016/j.swevo.2020.100752

[5] Peng, Y. (2025). A review of simulation optimization with connection to machine learning in warehouse management. *Computers & Industrial Engineering, 179*, 106076. https://doi.org/10.1016/j.cie.2024.106076

[6] Badakhshan, E. (2024). Application of simulation and machine learning in supply chain management. *Computers & Industrial Engineering, 179*, 106073. https://doi.org/10.1016/j.cie.2024.106073

[7] Griffioen, N. (2024). Efficient annotation reduction with active learning for computer vision-based retail product recognition. *Journal of Computational Social Science, 7*(2), 1039–1070. https://doi.org/10.1007/s42001-024-00266-7

[8] Chang, K. H. (2025). An integrated model for predictive maintenance and warehouse operations using simulation optimisation. *Computers & Industrial Engineering, 180*, 106085. https://doi.org/10.1016/j.cie.2024.106085

[9] Nicoletti, B. (2025). The impact of AI foundation models on the future of digital warehouse management. *Computers in Industry, 137*, 103582. https://doi.org/10.1016/j.compind.2025.103582

[10] Kahalimoghadam, M. (2025). An intelligent multi-agent system for last-mile logistics in warehouse operations. *Computers & Industrial Engineering, 180*, 106086.

https://doi.org/10.1016/j.cie.2024.106086ScienceDirect

[11] Bertsimas, D. (2024). A machine learning approach to two-stage adaptive robust optimisation in warehouse logistics. *Computers & Industrial Engineering, 179*, 106077. https://doi.org/10.1016/j.cie.2024.106077

[12] Rizqi, Z. U. (2024). Multi-objective simulation-optimisation for integrated automated storage and retrieval systems planning. *Computers & Industrial Engineering, 179*, 106074. https://doi.org/10.1016/j.cie.2024.106074

[13] Tang, H. (2025). Graph network surrogate model for optimising warehouse layout. *Computers & Industrial Engineering, 180*, 106087. https://doi.org/10.1016/j.cie.2024.106087

[14] Asadujjaman, M. (2024). Supply chain integrated resource-constrained multi-project scheduling problem: A warehouse perspective. *Computers & Industrial Engineering, 179*, 106075. https://doi.org/10.1016/j.cie.2024.106075

[15] Belter, J. (2023). Motion trajectory prediction in warehouse management using LSTM networks. *Applied Sciences, 13*(17), 9780. https://doi.org/10.3390/app13179780

[16] Sharifnia, S. M. E. (2021). Robust simulation optimisation for supply chain problem-solving. *Computers & Industrial Engineering, 157*, 107268. https://doi.org/10.1016/j.cie.2021.107268

[17] McSpadden, D. (2024). A comparison of machine learning surrogate models for warehouse process optimization. *Computers & Industrial Engineering, 179*, 106079. https://doi.org/10.1016/j.cie.2024.106079

[18] Hickman, X. (2025). Tackling distribution shift and outliers with the Student-t's distribution in warehouse data modelling. *Computers & Industrial Engineering, 180*, 106088. https://doi.org/10.1016/j.cie.2024.106088

[19] Villegas-Ch, W. (2024). Optimisation of inventory management through computer vision in warehouse settings. *Computers & Industrial Engineering, 179*, 106080. https://doi.org/10.1016/j.cie.2024.106080

[20] Ye, W. (2016). A computationally efficient simulation-based optimization method for warehouse inventory management. *Computers & Industrial Engineering, 97*, 1–12. https://doi.org/10.1016/j.cie.2016.05.001