

Article

A Deep Learning-Based Multi-Objective Optimization Method for Intelligent Maintenance Work Order Allocation

Lin Bai ¹, Jia Wang ^{1,*} and Xiaoping Zhou ^{2,*} 

¹ School of Intelligent and Technology, Beijing University of Civil Engineering and Architecture, Beijing 102616, China; 17797166577@163.com

² Beijing Key Laboratory of Intelligent Processing for Building Big Data, Beijing University of Civil Engineering and Architecture, Beijing 102616, China

* Correspondence: wangjia@bucea.edu.cn (J.W.); zhouxiaoping@bucea.edu.cn (X.Z.)

Abstract

The fire protection maintenance market is expanding rapidly, but the industry faces critical resource optimization challenges due to the inefficiency of existing scheduling methods. Current approaches often exhibit limited feature representation capabilities, inadequate handling of hard constraints (e.g., certification requirements), and poor balancing of multi-objective conflicts, compromising safety, compliance, and efficiency. To address these gaps, this paper proposes a deep learning-based multi-dimensional constraint feature intelligent dispatch algorithm (DL-MFIDA). The method constructs a deep feature learning framework to capture complex relationships between personnel skills and task requirements, incorporates a constraint-aware attention mechanism to strictly enforce hard constraints, and employs a multi-objective hierarchical optimization strategy to dynamically balance spatiotemporal cost minimization, skill matching gain maximization, and load balancing. Experimental results demonstrate that DL-MFIDA achieves an 89.5% assignment success rate even under high-load scenarios (task volume at five times personnel capacity), significantly outperforming traditional methods in key metrics such as allocation success rate, resource utilization, and constraint violation rate. This work provides an effective solution to the “few personnel, many tasks” dilemma in fire protection maintenance, ensuring robust performance in practical applications.

Keywords: building fire protection maintenance; intelligent dispatching; multi-objective optimization; deep learning

1. Introduction

The reliable operation of building fire protection facilities is paramount for safeguarding social assets and ensuring public safety, as these systems serve as the first line of defense during fire emergencies [1,2]. With the proactive implementation of China’s “14th Five-Year Plan for National Fire Protection Work,” the fire protection maintenance market has experienced substantial growth. However, this expansion occurs alongside intensified industry competition and stringent government price guidance [3], creating a pervasive operational dilemma for maintenance companies: the contradiction between achieving scale growth and maintaining efficiency. This often manifests as declining profitability despite increasing project volumes.

A primary driver of this inefficiency is the industry’s continued reliance on manual, experience-based dispatching for maintenance personnel within a “multi-project parallel”



Received: 24 January 2026

Revised: 23 February 2026

Accepted: 24 February 2026

Published: 26 February 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

operational model. This approach struggles to cope with dynamic workloads and complex, multi-constraint decision-making, leading to suboptimal resource allocation, increased operational costs, and potential compliance risks. Consequently, there is an urgent need for intelligent scheduling systems that can maximize workforce utilization—achieving the greatest possible workload with limited personnel—under strict professional skill requirements and time constraints.

Despite the critical nature of this problem, academic research specifically addressing personnel scheduling models for building fire protection maintenance remains scarce. Existing literature often prioritizes technical innovations on the equipment side. For instance, Rahardjo et al. [4] rightly emphasize that comprehensive maintenance must be performed by qualified personnel but stop short of exploring the decision-making mechanisms for efficiently scheduling those qualified personnel. While optimization methodologies from other domains (e.g., logistics, manufacturing) offer valuable insights, their direct application to the fire protection maintenance context reveals significant limitations:

1. **Limited Feature Representation:** Conventional methods typically depend on shallow, hand-crafted features, which fail to capture the complex, non-linear relationships between the multi-faceted skills of maintenance personnel and the specific requirements of diverse maintenance tasks, resulting in limited matching accuracy.
2. **Inadequate Constraint Handling:** Existing algorithms frequently employ simple rule-based filtering or soft-constraint penalty functions. These approaches cannot provide strict guarantees for satisfying hard constraints, such as mandatory professional certifications and domain-specific qualifications. This shortcoming is particularly critical in fire protection, where non-compliance can introduce significant safety hazards, especially for tasks involving electrical or mechanical systems.
3. **Inefficient Multi-Objective Balancing:** Traditional approaches often resort to weighted-sum scalarization to handle multiple objectives. This method is often inadequate for effectively balancing inherently conflicting goals, such as minimizing spatiotemporal costs, maximizing skill-matching gains, and ensuring equitable workload distribution across personnel.

These gaps collectively highlight the necessity for an integrated intelligent dispatching algorithm tailored to the building fire protection industry—one that simultaneously addresses feature representation, hard constraint satisfaction, and multi-objective optimization. To this end, our work is guided by the following research questions:

- (R1) How can deep learning be leveraged to learn expressive feature representations for employees and work orders, accurately capturing their intrinsic attributes and complex interrelationships?
- (R2) How can a constraint-handling mechanism be designed to ensure the strict satisfaction of hard constraints while providing optimization guidance for soft constraints?
- (R3) How can a multi-objective decision-making model be established to find a superior balance among conflicting objectives and maximize overall system efficiency? To address these challenges, this paper proposes a novel Deep learning-based Multi-dimensional constraint Feature-aware Intelligent Dispatch Algorithm (DL-MFIDA) for building fire protection maintenance. The key contributions of this work are fourfold:
 - We formulate the maintenance personnel assignment problem as a constrained multi-objective optimization model, explicitly incorporating domain-specific constraints from technical standards.
 - We develop a deep feature learning framework to extract high-level representations of both maintenance personnel capabilities and task requirements.

- We introduce a constraint-aware attention mechanism that strictly enforces hard constraints during the assignment process.
- We adopt a multi-objective hierarchical optimization strategy to dynamically balance the trade-offs between conflicting objectives like cost, skill matching, and load balancing.

Experimental results on synthetic datasets demonstrate that our proposed DL-MFIDA algorithm significantly outperforms traditional baseline methods across key performance indicators, including allocation success rate, resource utilization, and constraint violation rate. Notably, it maintains robust performance even under high-load scenarios, offering a practical solution to the “few personnel, many tasks” resource dilemma in the fire protection maintenance industry.

The remainder of this paper is organized as follows. Section 2 reviews related work in resource scheduling and deep learning for optimization. Section 3 details the problem formulation and the multi-objective optimization model. Section 4 presents the architecture and components of the proposed DL-MFIDA algorithm. Section 5 describes the experimental setup, results, and analysis. Finally, Section 6 concludes the paper and discusses future research directions.

2. Related Work

Our research intersects with two primary domains: classical resource scheduling methodologies and contemporary deep learning-based optimization techniques. This section provides a critical review of relevant literature from these fields, highlighting their contributions and limitations in addressing the specific challenges of intelligent dispatching in building fire protection maintenance.

2.1. Classical Approaches to Resource Scheduling

The problem of intelligently allocating maintenance personnel can be formally categorized as a Resource-Constrained Project Scheduling Problem (RCPSP) [5]. The core objective of RCPSP is to optimize the allocation of limited resources to activities over time, subject to various constraints. Existing solution strategies for RCPSP can be broadly classified into two categories: exact optimization methods and metaheuristic algorithms.

Exact optimization methods, such as Mixed-Integer Linear Programming (MILP) [6], dynamic programming, and constraint programming, are guaranteed to find the optimal solution for a given model. For instance, Van der Weide et al. [6] applied robust optimization techniques for long-term aircraft maintenance scheduling under uncertainty. Recent work by Mehr et al. [7] further developed interval-valued fuzzy optimization for multi-mode, multi-project scheduling under uncertainty, enhancing robustness. However, a fundamental limitation of these methods is their computational complexity, which grows exponentially with the problem size. This makes them intractable for the dynamic, medium-to-large-scale scheduling instances encountered in real-world fire protection maintenance scenarios, where rapid decision-making is required.

Metaheuristic algorithms, particularly swarm intelligence optimization, have been widely adopted as efficient alternatives for complex scheduling problems. These algorithms, inspired by natural phenomena, demonstrate strong global search capabilities and are less prone to becoming trapped in local optima compared to simple heuristics. Significant applications include the following: An interactive bacterial foraging optimization algorithm was developed by Niu et al. [8] for aircraft base maintenance personnel scheduling, improving base operational efficiency. Genetic Algorithms (GAs) have been adapted for task scheduling in unmanned testing systems [9] and for optimizing inspection routes for UAVs in gas fields [10]. Zhao et al. [11] also proposed an adaptive switching

algorithm based on improved ant colony optimization for resource scheduling in digital array radars, demonstrating the versatility of metaheuristics. Hybrid approaches, such as combining ant colony optimization with large-scale neighborhood search, have been proposed to enhance delivery personnel efficiency [12]. For multi-objective scenarios, Chen et al. [13] proposed an adaptive NSGA-II variant to improve the robustness of aircraft maintenance task schedules, and Zhang et al. [14] improved NSGA-III for Kubernetes resource scheduling.

Despite their successes, these model-driven methods exhibit critical shortcomings when applied to the fire protection maintenance domain. First, their inherent adaptability to multiple, strict hard constraints (e.g., professional certification matching) is often limited. They typically handle constraints via penalty functions, which cannot provide the strict guarantees required for safety-critical compliance. Second, their ability to balance multiple, conflicting objectives effectively is often inadequate. They struggle to adapt to the high-stakes environment of fire protection, where safety, compliance, and efficiency must be simultaneously prioritized, and they are generally not designed for the dynamic nature of real-time task arrival and personnel status updates.

2.2. Deep Learning for Optimization and Scheduling

Deep learning (DL), with its powerful capacity for automatic feature representation and complex relationship modeling, has opened new paradigms for solving combinatorial optimization problems, including scheduling [15]. By learning directly from data, DL-based methods can circumvent the need for meticulous hand-crafting of problem features and heuristics. Research in this area has explored various architectures:

Feature Enhancement: Nossayba et al. [16] integrated deep feature extraction with sentiment analysis to enhance the personalization capabilities of recommendation systems, demonstrating the utility of deep learning for capturing complex, latent features—a capability highly relevant to modeling the nuanced match between personnel skills and task requirements. Jing et al. [17] employed hypergraph convolution networks for efficient computing power resource scheduling, showcasing the power of advanced neural architectures for modeling complex relationships.

Deep Reinforcement Learning (DRL): Many studies have framed scheduling as a sequential decision-making process. Li et al. [18] proposed a DRL method (DRL-EPKG) that incorporates human knowledge into the reward function for autonomous driving decisions. Liang et al. [19] modeled online taxi dispatching as a Markov Decision Process (MDP), using reinforcement learning to decompose state value functions for real-time scheduling. Similarly, Xu et al. [20] developed methods based on proximal policy optimization (PPO) and predictive RL to address dynamic state representation and multi-objective coupling in complex systems. Recent applications are abundant: Zhang et al. [21] utilized a PPO algorithm for dual-resource scheduling of production equipment and vehicles; Qin et al. [22] designed an online learning-based resource scheduling scheme for satellite-terrestrial integrated networks; and Yan et al. [23] applied optimized adaptive density control in training resource scheduling for 3D Gaussian reconstruction. Li et al. [24] further advanced this frontier by integrating Large Language Models (LLMs) with fuzzy state fusion and reinforcement learning for dynamic multi-objective service resource scheduling.

Hybrid DL-Optimization Models: Another trend involves leveraging deep neural networks as surrogate models or within broader optimization frameworks. Similarly, Tipu et al. [25] and Komarasamy et al. [26] combined deep learning with multi-objective particle swarm optimization (MOPSO) for concrete mix design and cloud task scheduling, respectively, using DL for prediction and metaheuristics for searching the solution

space. This hybrid paradigm continues to evolve, as seen in Hosseini et al.'s study on [27] blockchain-enabled hybrid evolutionary scheduling for cloud resource optimization.

While these data-driven approaches excel in feature representation and adaptability, our analysis identifies a persistent application gap concerning the fire protection maintenance domain. Specifically, existing DL methods often lack explicit, guaranteed mechanisms for handling hard constraints, treating them as soft penalties within the learning objective. This is a critical deficiency in a field governed by stringent safety regulations. Furthermore, the generalization ability of these models can significantly degrade under the strict and non-negotiable constraints present in our target scenario. Many models achieve good performance in specific, often simplified settings but lack the robustness for environments with multiple, concurrent task types and dynamic, hard constraint sets.

2.3. Cross-Domain Applications and Broader Context

Recent research highlights the expanding application scope of resource scheduling methodologies. In networked systems, Yang et al. [28] addressed constrained multi-agent resource optimization with channel prediction in 6G non-terrestrial networks, while V et al. [29] focused on energy-efficient resource scheduling for optical access networks. In cyber-physical systems, Zhao et al. [30] developed a collaborative metaverse-digital twin system for traffic perception and resource scheduling. For large-scale infrastructure, Gao et al. [31] presented a multi-scale multi-objective optimization model for water resource scheduling. Cloud computing environments have also seen sustained innovation, with S et al. [32] proposing a multi-objective optimization-based efficient resource scheduling scheme for IaaS clouds. These studies demonstrate the universal challenge of efficient resource allocation across disparate domains but also underscore the unique constraints and objectives inherent to each specific field, like the stringent safety and certification requirements in fire protection maintenance that are not primary concerns in computational or network resource scheduling.

2.4. Positioning of Our Work

The review of existing literature clearly situates our contribution. Traditional optimization methods provide a solid mathematical foundation but are often too rigid or computationally expensive. Modern deep learning methods offer flexibility and powerful pattern recognition but are unreliable for strict constraint satisfaction. Our proposed DL-MFIDA algorithm is designed to bridge this gap. It leverages the representational power of deep learning to model the complex features of personnel and tasks while introducing a novel constraint-aware attention mechanism to strictly enforce hard constraints. Furthermore, it incorporates a multi-objective hierarchical optimization strategy to dynamically balance the trade-offs between conflicting objectives like spatiotemporal cost, skill matching, and load balancing. This hybrid approach aims to achieve the robustness of classical methods with the adaptability and representational power of deep learning, tailored specifically to the critical needs of building fire protection maintenance scheduling.

3. A Multi-Objective Optimization Model for Matching Decisions

This section formalizes the intelligent dispatching problem for building fire protection maintenance personnel into a constrained multi-objective optimization framework. The model is designed to support decision-making in a dynamic environment with multiple concurrent tasks, ensuring that allocations are not only efficient but also strictly compliant with operational regulations and safety standards.

3.1. Problem Description and Modeling Foundations

The model construction is grounded in the practical requirements of fire maintenance work, particularly drawing from standards such as Clause 6.1.2.2 of the Technical Specification for Maintenance of Building Fire Protection Facilities (DB11/T 3035-2023) [33]. The core elements extracted from maintenance task attributes, as exemplified in Figure 1, are mapped into the model as follows:

1. **Task Attribute Elements:** Key attributes—including task type, equipment category, maintenance cycle, and specific maintenance content—are transformed into qualitative and quantitative constraints. Qualitatively, the task type determines the required professional domain (e.g., electrical, plumbing, HVAC) and corresponding mandatory certification requirements (e.g., electrician’s license). Quantitatively, the maintenance content dictates the minimum proficiency threshold for personnel and the task’s completion deadline.
2. **Personnel Capability Elements:** In accordance with task demands, maintenance personnel are characterized by their professional domains, held certifications, historical task proficiency, current workload, and availability.
3. **Environmental Constraints:** Time window constraints are established based on the mandated maintenance cycles, and personnel workload is capped to remain within reasonable limits, preventing burnout and ensuring quality.

6.1 Fire Protection Power Supply and Distribution Facilities

6.1.2 Inspection Content

6.1.2.2 Every six months a functional test of the automatic transfer capability for the dual power automatic transfer switch shall be conducted.

- **Under automatic control mode:** Manually cut off the main fire protection power supply and observe the engagement of the standby fire protection power supply and the status of indicator lights.
- **Under manual control mode:** In the low-voltage distribution room, first cut off the main fire protection power supply, then close the standby fire protection power supply, and observe its engagement and the status of indicator lights.

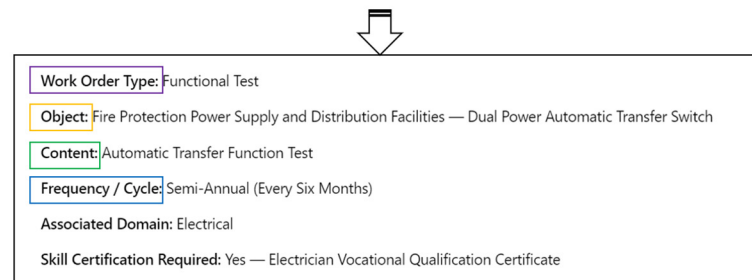


Figure 1. Example of maintenance tasks.

The problem is essentially a matching problem between a set of tasks and a set of employees. Its mathematical essence is to find the optimal task–employee assignment by optimizing a set of objective functions—such as skill matching degree, load balance, and response time—under a set of strict constraints. To simplify the model expression and enhance its computational feasibility, we introduce the following reasonable assumptions:

- **Task Type Assumption:** All considered maintenance tasks are planned (e.g., monthly or quarterly inspections) with pre-determined execution time windows. Emergency repair orders are not considered in this stage of the model.
- **Cost Conversion Assumption:** The costs associated with employee geographical location and route planning are uniformly quantified into temporal costs for standardized processing.

The complete mathematical model framework, which integrates personnel modeling, task modeling, constraint expression, and multi-objective function construction, is visually summarized in Figure 2 below.

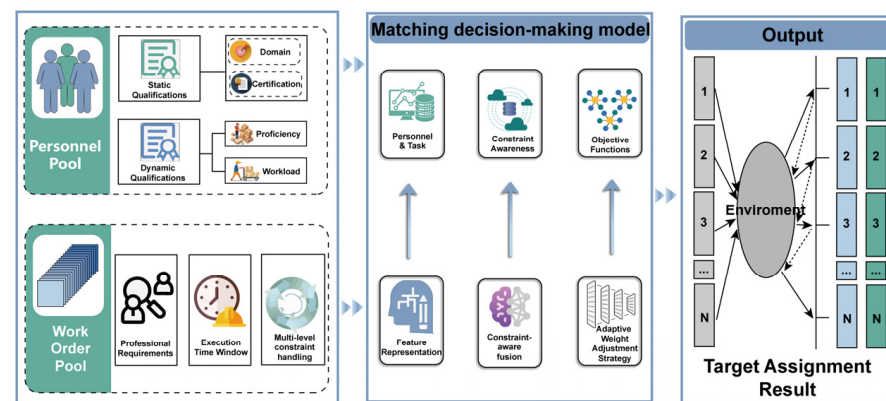


Figure 2. Framework Diagram.

3.2. Personnel and Task Modeling

As the primary executors of tasks, the professional qualifications and current status of maintenance personnel directly impact the quality and timeliness of task completion. Therefore, we abstract each employee $i \in \mathcal{I}$ as a quintuple: $E_i = (k_i, p_i, q_i, w_i, t_i)$ where:

- $k_i = \{(k, c_k) | k \in \mathcal{K}, c_k \in \{0, 1\}\}$ represents the set of professional fields that the employee is proficient in, along with a binary indicator showing whether a valid certificate is held for that field (1 if yes, 0 otherwise).
- $p_i \in [0, 1]$ denotes the employee's historical proficiency rating in their professional domain tasks.
- $q_i \in [0, 1]$ signifies the employee's historical task completion quality score.
- $w_i \in [0, 1]$ is the ratio of the employee's current unfinished task duration to their maximum workload capacity $L_i \in \mathbb{N}^+$.
- $t_i \in \mathbb{R}^+$ represents the employee's last assignment time or the current system time.

This multi-faceted representation effectively balances the static qualifications and dynamic status of employees, facilitating both rigorous constraint checking and flexible evaluation of execution efficiency.

Similarly, fire protection maintenance tasks have explicit professional requirements, execution deadlines, and skill thresholds. Each task $j \in \mathcal{J}$ is defined as a quintuple: $O_j = (k_j, r_j, d_j, u_j, t_j)$ where:

- $k_j = \{(k, c_k) | k \in \mathcal{K}, c_k \in \{0, 1\}\}$ specifies the required professional field and a binary indicator for mandatory certification (1 if required, 0 otherwise).
- $r_j \in [0, 1]$ is the minimum skill proficiency level required from an employee to undertake the task.
- $d_j \in \mathbb{R}^+$ is the remaining response time for the task.
- $u_j \in \mathbb{R}^+$ is the estimated time required to complete the task.
- $t_j \in \mathbb{R}^+$ is the latest permissible completion time for the task.

These attributes are primarily used to filter the set of eligible employees and to influence the weighting within the multi-objective function. A comprehensive description of all parameters is provided in Table 1.

Table 1. Description of attribute parameters used in the mathematical model.

Symbol	Domain	Describe
$\mathcal{I}$	$\{1, 2, \dots, N\}$	Set of N employees.
$\mathcal{K}$	$\{1, 2, \dots, K\}$	Set of K professional fields (e.g., electrical, plumbing).
$\mathcal{J}$	$\{1, 2, \dots, M\}$	Set of M tasks to be assigned.
k_i	$\mathcal{K}$	Professional field required for employee i .
c_k	$\{0, 1\}$	Certificate holding status for a specific field (1: holds certificate).
p_i	$[0, 1]$	Employee's proficiency rating in their professional domain.
q_i	$[0, 1]$	Employee's task completion quality score.
w_i	$[0, 1]$	Employee's current workload ratio.
L_i	$\mathbb{N}^+$	Employee's maximum workload capacity (total working hours).
t_i	$\mathbb{R}^+$	Employee's last assignment time or current system time.
k_j	$\mathcal{K}$	Professional field required for task j .
c_j	$\{0, 1\}$	Indicator if task j requires a professional certificate (1: required).
r_j	$[0, 1]$	Minimum skill level required from an employee for task j .
d_j	$\mathbb{R}^+$	Remaining response time for task j .
u_j	$\{0, 1\}$	Estimated time required to complete task j .
t_j	$\mathbb{R}^+$	Latest completion time for task j .

3.3. Constraint Formulation

The proposed allocation scheme must satisfy the following five hard constraints, formulated in Equations (1)–(5):

$$\sum_{i \in \mathcal{I}} x_{ij} \leq 1, \forall j \in \mathcal{J} \quad (1)$$

$$\sum_{j \in \mathcal{J}} u_j \cdot x_{ij} \leq L_i - w_i, \forall i \in \mathcal{I} \quad (2)$$

$$x_{ij} = 0, \forall (k, c_k) \in k_i \text{ s.t. } (k = k_j \wedge c_k \geq c_j) \quad (3)$$

$$x_{ij} = 0, \text{ if } p_i < r_j \quad (4)$$

$$x_{ij} = 1, \text{ if } t_i + T_{ij} \leq t_j - u_j \quad (5)$$

where Equation (1) is the unique allocation constraint, indicating that each work order can be executed by at most one employee; Equation (2) is the personnel load constraint, meaning that the number of tasks each employee undertakes at the same time cannot exceed their remaining load capacity; Equation (3) is the professional qualification constraint, ensuring that employees meet both the professional field and certification requirements if the task requires a certificate; Equation (4) is the skill threshold constraint, meaning that the personnel's proficiency must meet the task requirements; Equation (5) is the spatiotemporal constraint: this paper uniformly converts the geographical location and path cost factors of employees in the order dispatch process into time costs, and the employee's arrival time must be earlier than the latest task completion time. x_{ij} is the decision variable; if employee i is assigned to task j , then $x_{ij} = 1$, otherwise 0, and the remaining parameters are shown in Table 1.

3.4. Objective Functions

The optimization objectives of this study primarily encompass three critical dimensions: spatiotemporal cost minimization, skill matching gain maximization, and load balancing optimization. These objectives are designed to address the fundamental challenges in fire maintenance dispatch operations, balancing operational efficiency, quality

assurance, and resource sustainability. The mathematical formulations are established as follows:

$$f_1 = \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} x_{ij} \cdot T_{ij} \quad (6)$$

$$f_2 = \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} x_{ij} \cdot \max(0, p_i - r_j) \quad (7)$$

$$f_3 = \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} x_{ij} \cdot w_i \quad (8)$$

In this formulation, f_1 represents the minimization of total spatiotemporal cost, quantifying the efficiency of task allocation across geographical and temporal dimensions. The parameter T_{ij} encompasses both travel time and task execution duration, providing a comprehensive cost metric. The second objective, f_2 , aims to maximize skill matching gain by promoting the assignment of work orders to employees whose proficiency (p_i) exceeds the task's minimum requirement (r_j). This ensures optimal task-personnel compatibility while maintaining quality standards. The $\max(0, p_i - r_j)$ term acts as a filtering mechanism, generating positive utility only when personnel competency meets or surpasses task demands. The third objective, f_3 , focuses on load balancing optimization by penalizing assignments to employees approaching their capacity limits (w_i), thereby preventing resource overutilization and promoting sustainable workforce management.

To transform this multi-objective optimization problem into a solvable single-objective framework, we employ the weighted sum method, a widely adopted approach in operational research for managing conflicting objectives. The normalized objective function is formulated as:

$$\begin{aligned} \min_{x_{ij}} Z &= \alpha \sum_{i,j} \frac{x_{ij}}{d_j + \epsilon} + \beta \left(- \sum_{i,j} x_{ij} \cdot \max(0, p_i - r_j) \right) + \gamma \sum_{i,j} x_{ij} \cdot w_i \\ \text{s.t. } &\sum_i x_{ij} \leq 1, \forall j \in \mathcal{J} \\ &\sum_j x_{ij} \leq L_i, \forall i \in \mathcal{I} \\ &x_{ij} = 0, \text{ if } c_j = 1 \wedge (k_i \neq k_j \vee c_i = 0) \\ &x_{ij} = 0, \text{ if } p_i < r_j \\ &x_{ij} \in \{0, 1\}, \forall i, j \end{aligned} \quad (9)$$

The weighting parameters $\alpha, \beta, \gamma \in [0, 1]$ satisfy the normalization condition, representing the relative importance assigned to each objective function. The parameter serves as a small constant to ensure numerical stability in the optimization process. This integrated formulation enables systematic trade-off analysis among the competing objectives while maintaining computational tractability for large-scale practical applications.

4. Deep Learning-Based Multi-Dimensional Constraint-Aware Feature-Driven Intelligent Dispatch Algorithm (DL-MFIDA)

Traditional deep learning methods (e.g., DPFLE) have certain defects when dealing with such problems. For instance, static decision-making defects cause the model to be unable to respond to dynamic environmental changes (e.g., changes in employee load, location movement). When any attribute changes, the complete inference process needs to be rerun, causing unacceptable delay. Crude constraint handling mechanisms mean that existing gated attention mechanisms only achieve soft constraint filtering; although they can reduce the probability of constraint violation, they cannot provide strict guarantees. This shortcoming may lead to serious consequences in critical scenarios like fire protection.

4.1. Algorithm Framework Design

Addressing the challenges faced by the above model in practical solving, this paper proposes a Deep Learning-based Multi-constraint Feature-driven Intelligent Dispatch Algorithm (DL-MFIDA). By transforming the traditional optimization problem into a representation learning problem under a deep learning framework, effective connection between the mathematical model and computational method is achieved. The core architecture of the algorithm includes three key modules: the feature encoding module is responsible for converting the raw attributes of maintenance personnel and tasks into distributed representations; the constraint processing module ensures the solution space satisfies all hard constraints; the multi-objective optimization module balances multiple conflicting objectives through an adaptive weight mechanism. Figure 3 shows the overall workflow of the algorithm, reflecting the complete computational path from raw data to final decision.

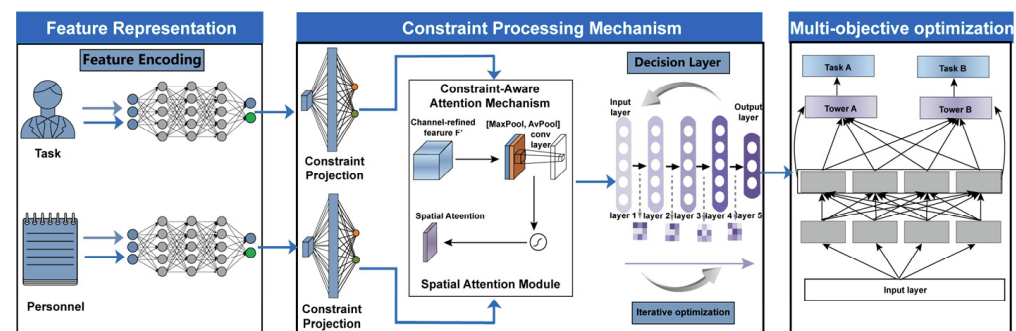


Figure 3. Algorithm Flowchart.

4.2. Feature Representation Learning

In the feature encoding stage, the algorithm uses deep neural networks to automatically learn the feature representations of personnel and tasks. For each maintenance employee $i \in \{1, 2, \dots, N\}$, their feature representation is:

$$h_i = f_\theta(e_i) \quad (10)$$

where $e_i = [k_i, c_k, p_i, q_i, w_i, L_i, t_i]^T$ is the raw feature vector of personnel i , and f_θ is a parameterized deep neural network. For each task $j \in \{1, 2, \dots, M\}$, its feature representation is:

$$g_j = g_\phi(o_j) \quad (11)$$

where $o_j = [k_j, c_j, r_j, d_j, u_j, t_j]^T$ is the raw feature vector of task j , and g_ϕ is a parameterized deep neural network.

Architecture Design of Feature Encoders: Both the employee feature encoder f_θ and the task feature encoder g_ϕ are implemented using multilayer feedforward neural networks with identical structures but independently trained parameters. For categorical fields in the input (e.g., professional domains k_i and k_j), we first map them to 16-dimensional continuous vectors through trainable embedding layers. These embedding vectors are subsequently concatenated with numerical features (e.g., proficiency p_i) to form complete input tensors.

The specific network architecture is configured as follows: an input layer receives the concatenated feature vectors, and two fully connected hidden layers with dimensions of 128 and 64 employ ReLU activation functions to introduce nonlinear transformations. The ReLU function is defined as:

$$\text{ReLU}(x) = \max(0, x) \quad (12)$$

This activation function is chosen for its advantages in mitigating the vanishing gradient problem and promoting sparse activations, which enhances the network's ability to capture complex feature interactions. Finally, an output layer with dimension 32 generates the ultimate feature representations h_i and g_j . This design ensures representational capacity while effectively controlling model complexity.

Training Configuration and Hyperparameter Optimization: Model training is guided by the minimization of a multi-objective weighted loss function, which directly corresponds to the objective function Z defined in Section 3.4. The specific formulation is:

$$Loss = \alpha \cdot Norm(f_1) - \beta \cdot Norm(f_2) - \gamma \cdot Norm(f_3) \quad (13)$$

where $Norm(\cdot)$ denotes min-max normalization employed to balance scale differences among different objectives. The loss weights α, β, γ are empirically determined through grid search.

Training utilizes the Adam optimizer with an initial learning rate of 0.001, combined with an exponential decay strategy (decay rate 0.95, applied every 50 epochs). The batch size is fixed at 32 to ensure stable gradient estimation within memory constraints. We implement early stopping to prevent overfitting: training terminates if the validation loss fails to improve for 10 consecutive epochs, with the maximum number of epochs capped at 200.

Hyperparameter Selection Process: Critical hyperparameters were determined through grid search within the following spaces: hidden layer dimensions {64, 128, 256}, learning rates {0.01, 0.001, 0.0001}, and batch sizes {16, 32, 64}. The final hyperparameter combination, selected based on multi-objective performance on the validation set, is summarized in Table 2.

Table 2. Hyperparameter configuration for feature encoders.

Hyperparameter	Search Space	Final Value
Hidden Layer 1 Dimension	[64,128,256]	128
Hidden Layer 2 Dimension	[32,64,128]	64
Embedding Dimension	[8,16,32]	16
Initial Learning Rate	[0.01,0.001,0.0001]	0.001
Batch Size	[16,32,64]	32
Early Stopping Patience	[5,10,15]	10

Through the aforementioned encoding process, the algorithm effectively transforms the raw feature vectors into semantic feature representations, as defined in Equations (10) and (11). Meanwhile, the ReLU-activated hidden layers enable the network to learn hierarchical feature representations, where lower-level features are combined to form more abstract higher-level representations. This provides high-quality input for the subsequent constraint-aware attention mechanism and multi-objective decision-making process.

4.3. Constraint Processing Mechanism

To ensure the generated decision scheme satisfies all hard constraints, the algorithm designs a multi-level constraint processing mechanism. First, constraint projection is performed in the feature space by introducing a constraint-aware attention mechanism, as in Equation (14):

$$\alpha_{ij} = \text{softmax}\left(\frac{\phi_i^T W \phi_j}{\sqrt{d}}\right) \cdot \Pi[C(i, j)] \quad (14)$$

where $\text{softmax}(\frac{\phi_i^T W \varphi_j}{\sqrt{d}})$ is used to calculate the matching degree in the feature space, and $\Pi[C(i, j)]$ is an indicator function that takes the value 1 when the personnel i satisfies all constraints for task j , and 0 otherwise. Secondly, a constraint correction module is introduced at the decision layer, as in Equation (15), ensuring solution feasibility through iterative optimization:

$$X^{(t+1)} = \text{proj}_F(X^{(t)} - \eta \nabla f(X^{(t)})) \quad (15)$$

where proj_F denotes the projection operation onto the feasible region. The feasible set projection operator proj_F maps any assignment matrix X to the feasible solution space F that satisfies all hard constraints. For projecting binary assignment matrices under multiple coupled constraints, we employ an iterative repair algorithm, the details of which are shown in Algorithm 1.

Algorithm 1. Multi-Constraint Projection Algorithm

Input: Initial assignment matrix $X^{(0)}$, maximum iterations $T_{\max}$

Output: Feasible assignment matrix X^*

1. Initialization: Set $t = 0$, $X^{(0)}$ as the matrix to be projected
 2. Constraint Satisfaction Check: Sequentially verify satisfaction of constraints (1)–(5)
 3. Iterative Repair:
 - For tasks violating unique assignment constraint (1), maintain the assignment with highest weight and set others to zero
 - For personnel violating load constraint (2), retain tasks in descending order of urgency, setting overloaded tasks to zero
 - Qualification constraint (3) and skill constraint (4) are ensured through pre-filtering and do not participate in iterative repair
 - Spatiotemporal constraint (5) is guaranteed through feasibility checks during assignment
 4. Convergence Check: If all constraints are satisfied or $t > T_{\max}$, proceed to step 5; otherwise set $t = t + 1$ and go to step 2
 5. Output: Return feasible solution $X^* = X^{(t)}$
-

The projection algorithm has a time complexity of $O(T_{\max} \times m \times n)$, where m and n represent the number of personnel and tasks, respectively, and $T_{\max}$ denotes the maximum allowed iterations. The algorithm is guaranteed to terminate with a feasible solution since each iteration resolves at least one constraint violation, and the finite constraint set ensures bounded convergence.

It should be noted that the aforementioned mechanisms ensure strict satisfaction of hard constraints. However, in the pursuit of multi-objective optimization, the algorithm may generate intermediate solutions that slightly violate soft constraints (e.g., the load constraint). This is because our multi-objective hierarchical optimization strategy (see Section 4.4) allows temporary acceptance of minor deviations from soft constraints during the iterative process to explore a better balance of the objective functions. The final decision solution is pulled back to the feasible region satisfying all hard constraints through the projection operation proj_F in Equation (15), but for soft constraints, we focus more on achieving an overall optimum among conflicting objectives rather than absolute zero violation. This is exactly what the CVR metric in the experimental section reflects—a multi-objective trade-off result under the guarantee of strict hard constraint satisfaction.

4.4. Multi-Objective Optimization Strategy

For the multi-objective optimization problem defined in the model, the algorithm employs an enhanced weighted sum method coupled with an adaptive weight adjustment mechanism. This strategy dynamically balances the three competing objectives: spatiotemporal cost minimization (f_1), skill matching gain maximization (f_2), and load balance optimization (f_3). The composite objective at iteration is formulated as:

$$\min_X J^{(t)}(X) = \sum_{k=1}^3 \lambda_k^{(t)} f_k(X) \quad (16)$$

where $f_k(X)$ are the objective functions from Section 2.3, and the weight vector $\lambda^{(t)} = [\lambda_1^{(t)}, \lambda_2^{(t)}, \lambda_3^{(t)}]^T$ is iteratively refined based on the gradient information of each objective function, ensuring the search direction responds to the current optimization landscape. The update rule is formally defined in two steps to guarantee non-negativity and normalization:

1. Gradient-guided intermediate update:

$$\tilde{\lambda}_k^{(t+1)} = \lambda_k^{(t)} \cdot \exp(\eta g_k^{(t)}), \text{ where } g_k^{(t)} = \left. \frac{\partial f_k}{\partial X} \right|_{X^{(t)}} \quad (17)$$

2. Normalization:

$$\lambda_k^{(t+1)} = \frac{\tilde{\lambda}_k^{(t+1)}}{\sum_{j=1}^3 \tilde{\lambda}_j^{(t+1)}} \quad (18)$$

where $\eta > 0$ is a learning rate. This design inherently enforces the constraints $\lambda_k^{(t)} > 0$ and $\sum_{k=1}^3 \lambda_k^{(t)} = 1$ for all t , maintaining a valid convex combination of the objectives throughout the optimization.

This adaptive mechanism facilitates a dynamic exploration of the objective space. When an objective (e.g., f_1) has significant potential for improvement (large gradient magnitude), its weight increases, steering the solution towards regions that enhance that specific objective. The normalization step prevents any single weight from dominating indefinitely, forcing a trade-off among objectives. Consequently, the algorithm navigates along the Pareto front, seeking a diverse set of non-dominated solutions that represent different optimal compromises between cost, quality, and fairness, rather than converging to a single point.

The update rule possesses inherent stability: (1) Boundedness: The exponential transformation and subsequent normalization confine all weights to the interval (0, 1). (2) Smooth guidance: The learning rate moderates the adjustment magnitude, preventing oscillatory behavior. In the context of fire maintenance dispatch, this strategy allows the optimization focus to shift intelligently based on the current state. For instance, during periods of high task backlog, the weight for spatiotemporal cost (f_1) may increase to prioritize efficiency. Conversely, if skill mismatches are detected, the weight for skill matching gain (f_2) can rise to ensure work quality, while the load balance weight (f_3) ensures equitable workload distribution.

To provide a clear overview of the DL-MFIDA workflow and enhance reproducibility, Algorithm 2 presents its structured pseudocode.

Algorithm 2. DL-MFIDA

Input: Personnel set $\mathcal{I}$, task set $\mathcal{J}$, personnel features $\{e_i\}_{i \in \mathcal{I}}$, task features $\{o_j\}_{j \in \mathcal{J}}$, maximum iterations T , learning rate η .

Output: Optimal assignment matrix X^*

1. Initialize: Iteration counter $t \leftarrow 0$, feature encoder parameters θ, ϕ , multi-objective weights $\lambda^{(0)} \leftarrow (1/3, 1/3, 1/3)$
2. Repeat
3. Step 1: Feature Representation Learning. Compute feature representations for all personnel and tasks using encoders f_θ and g_ϕ (Equations (10) and (11))
4. Step 2: Constraint-Aware Matching Score Calculation. For each pair (i, j) , compute the constraint-aware attention score (Equation (12))
5. Step 3: Multi-Objective Decision Generation. Generate candidate $X^{(t)}$ assignment by optimizing the weighted-sum loss (Equation (16)) with current weights $\lambda^{(t)}$
6. Step 4: Feasible Solution Projection. Project $X^{(t)}$ onto the feasible region F satisfying all hard constraints Equation (13) and Algorithm 1
7. Step 5: Adaptive Multi-Objective Weight Update. Update weights based on the performance of the current solution (Equation (17)):
8. Step 6: Iteration and Termination. $t \leftarrow t + 1$
9. Until $t > T$ or weights converge
10. Return optimal assignment matrix $X^* = \tilde{X}^{(t)}$

5. Experimental Design and Analysis

This section presents a comprehensive experimental evaluation to validate the effectiveness and superiority of the proposed algorithm. Performance comparisons are conducted from multiple perspectives against traditional algorithms under various scenarios, followed by an in-depth analysis of the results.

5.1. Experimental Setup

Hardware Environment: Intel(R) Core(TM) i7-13620H processor, 16GB RAM, NVIDIA GeForce RTX 4050 Laptop GPU.

Software Environment: Python 3.9, PyTorch 1.12, Flask 2.3.

Given the current lack of publicly available datasets for building fire maintenance work orders, this experiment employs a parametric synthetic dataset generated based on the statistical characteristics of the real-world BF-Open-Data [34]. The synthetic data generator is designed to rigorously simulate the joint distribution, key constraints, and intrinsic correlations of task and personnel attributes from real operational scenarios, ensuring the external validity and reproducibility of the experiments.

The generation process strictly adheres to the mathematical model defined in Section 3. Work Order Generation involves the following steps: task types (e.g., electrical inspection, water system maintenance, alarm debugging) and their corresponding mandatory professional certificate requirements are sampled from a multinomial distribution derived from BF-Open-Data statistics; the skill requirement threshold r_j follows a truncated normal distribution $N_{[0,1]}(\mu = 0.65, \sigma = 0.12)$, modeling that most maintenance tasks require medium-to-high skill levels; the estimated task duration u_j follows a Gamma distribution with shape parameter $k = 2.2$ and scale parameter $\theta = 1.3$ (in hours), with its right-skewed nature aligning with actual work order duration patterns; task locations are randomly generated on a simulated urban grid (100×100 units), and their latest completion time t_j is calculated based on the task generation timestamp, a preset urgency level (High, Medium, Low), and corresponding maximum response time limits (e.g., 2, 8, 24 h). Maintenance

Personnel Generation includes the following: each personnel's set of professional domains k_i is sampled from a skewed distribution to ensure a realistic ratio of specialists to multi-skilled workers; their held certificates c_k strictly correspond to their domains; historical proficiency p_i follows a Beta distribution $Beta(\alpha = 7, \beta = 2)$, and quality score q_i follows $Beta(\alpha = 7, \beta = 1.5)$, reflecting the positive skew in personnel skill levels; the daily maximum standard workload L_i is uniformly distributed in the interval [7,9] hours.

To simulate dependencies between attributes in the real world, the generator incorporates preset correlations: (1) Tasks with high skill requirements ($r_j > 0.8$) have an 80% probability of being labeled “High Urgency,” and their conditional expected duration is increased by 20%; (2) Personnel holding advanced or special operation certificates receive a baseline increase of 0.1 in their proficiency p_i for that domain. All distribution parameters (means, variances, correlation strengths) are iteratively calibrated via Maximum Likelihood Estimation against anonymized aggregate statistics from BF-Open-Data (e.g., task type proportions, average handling time, certificate coverage) until the deviation of key first- and second-order statistics of the synthetic data from the real source is less than 5%.

Table 3 presents a detailed statistical summary of the synthetic dataset for the benchmark scenario (500 tasks, 100 personnel). To evaluate algorithm robustness, in the scalability experiments of Section 5.4, we systematically scale the number of tasks and personnel proportionally and adjust parameters like load ratio and urgent task proportion. Each experimental scenario is independently executed 20 times using fixed random seeds, and the average performance metrics with 95% confidence intervals are reported.

Table 3. Synthetic dataset parameters and statistical summary (Benchmark: 500 tasks, 100 personnel).

Attribute	Distribution/Generation Rule	Parameter/Mean	Std. Dev./Range
Task Attributes			
Skill Req. r_j	Truncated Normal	$\mu = 0.65$	$\mu = 0.65$ [0.25, 0.98]
Duration u_j (hours)	Gamma	$k = 2.2, \theta = 1.3$	Mean 2.86, [0.75, 10.2]
Urgent Task Ratio	Multinomial Sampling	20% (High), 50% (Med), 30% (Low)	-
Cert.-Required Task Ratio	Conditional Probability	35%	-
Personnel Attributes			
Proficiency p_i	Beta	$\alpha = 7, \beta = 2$	Mean 0.78, [0.45, 0.99]
Max Load L_i (hours/day)	Uniform	-	[7, 9]
Certified Personnel Ratio	Conditional Sampling	85%	-
Correlated Attributes			
High-Skill & High-Urgency Link	Conditional Probability	$P(HighUrgency r_j > 0.8) = 0.8$	-
Certificate-Proficiency Gain	Additive Offset	$\Delta p_i = +0.1$ (if certified)	-

5.2. Evaluation Metrics

To comprehensively assess algorithm performance, four evaluation metrics were adopted:

1. Assignment Success Rate (ASR): The ratio of successfully assigned tasks to the total number of tasks, reflecting the fundamental allocation capability of the algorithm:

$$ASR = \frac{N_{assigned}}{N_{total}} \times 100\% \quad (19)$$

This metric calculates the proportion of tasks with “Assigned” status to all pending tasks within the statistical period.

2. Average Response Time (ART): The average time interval from task creation to the start of processing by an employee, indicating the algorithm's response speed:

$$ART = \frac{1}{N} \sum_{i=1}^N (t_{start}^i - t_{create}^i) \quad (20)$$

where only successfully assigned and initiated tasks are considered, excluding those waiting or unassigned.

3. Resource Utilization Rate (RUR): The ratio of actual working time to total available time for maintenance personnel, measuring human resource efficiency:

$$RUR = \frac{1}{M} \sum_{j=1}^M \frac{\omega_j}{L_j} \times 100\% \quad (21)$$

This metric helps prevent resource idling or overloading.

4. Constraint Violation Rate (CVR): This metric measures the proportion of assignments that violate soft constraints. As described in Section 3.3, our algorithm strictly guarantees the satisfaction of all hard constraints (e.g., certification requirements) through the constraint-aware attention mechanism and decision-layer projection. Therefore, the CVR reported in the experiments specifically refers to violations of soft constraints (primarily the personnel load constraint, i.e., Equation (2)) that may occur during the optimization process to balance multiple objectives. It is calculated as the ratio of the number of assignments violating any soft constraint to the total number of assignments:

$$CVR = \frac{N_{violation_soft}}{N_{assigned}} \times 100\% \quad (22)$$

5.3. Baseline Algorithms

To conduct a rigorous and comprehensive evaluation of the proposed DL-MFIDA algorithm, we selected and implemented four representative baseline algorithms for comparison: a classical heuristic (Genetic Algorithm, GA), an exact mathematical programming solver (Mixed-Integer Linear Programming, MILP), an advanced multi-objective evolutionary algorithm (NSGA-III), and a modern deep reinforcement learning-based scheduler (Deep Policy Network, DPN). To ensure a fair comparison, all baseline algorithms are encoded with the same mathematical model as our problem and share identical preprocessing (e.g., hard constraint filtering) and evaluation pipelines. Each algorithm underwent independent parameter tuning and was allocated a comparable computational time or training resource budget for iterative or learning-based methods.

5.3.1. Genetic Algorithm (GA)

As a classic metaheuristic optimization method, the Genetic Algorithm is widely used for solving complex combinatorial optimization problems. We implemented a customized GA for our specific problem. Its chromosome uses a binary encoding corresponding to the personnel–task matching relationships. The initial population is generated via a random heuristic that satisfies all hard constraints. The fitness function is defined as the multi-objective weighted sum $J(X)$ from Equation (16). Tournament selection is employed for the selection operator, uniform crossover for the crossover operator, and bit-flip mutation for the mutation operator. To ensure a fair comparison, this algorithm uses the same training/testing dataset split as DL-MFIDA and is evaluated on the test set.

The key parameters of the GA were determined through a systematic grid search on a held-out validation set. The search ranges were: population size $\in [50, 100, 200]$, number of generations $\in [100, 200, 500]$, crossover probability $\in [0.6, 0.7, 0.8, 0.9]$, and

mutation probability $\in [0.01, 0.05, 0.1]$. The performance was evaluated based on the average improvement of the composite objective function z on the validation set. The final configuration that provided stable convergence and the best performance across most scenarios was: a population size of 100, 200 generations, a crossover probability of 0.8, and a mutation probability of 0.05.

5.3.2. Mixed-Integer Linear Programming Solver (MILP)

To evaluate the theoretical optimal bound of the problem, we formulated the mathematical model from Section 3 as a Mixed-Integer Linear Programming problem and solved it using the commercial solver Gurobi (version 10.0). The MILP model directly optimizes the objective function in Equation (16) while strictly adhering to all constraints in Equations (1)–(5). Considering the NP-hard nature of the problem and the need for a practical runtime in scheduling applications, we set a computational time limit of 300 s for Gurobi for each instance. This limit provides a reasonable trade-off between solution quality and computational expense, aligning with near-real-time decision-making requirements. All MILP experiments were run under identical hardware conditions with default parallel computing settings (the number of threads was set to 8). In the experimental section, we report the best feasible solution found within this time limit.

5.3.3. NSGA-III Algorithm (NSGA-III)

NSGA-III is a reference point-based multi-objective evolutionary algorithm particularly suitable for optimization problems with three or more objectives, effectively maintaining the diversity of the solution set along the Pareto front. We implemented the standard NSGA-III framework. Its chromosome encoding is the same as that of the Genetic Algorithm. The algorithm uses simulated binary crossover and polynomial mutation operators to generate offspring.

The parameter setting follows the algorithm's design principles. First, based on the 91 structured reference points generated for our three objectives using the Das and Dennis method, the population size was set to 210. This size, which is approximately 2–3 times the number of reference points, is a common practice to ensure adequate population diversity and effective association with each reference direction. The number of generations and the distribution indices for crossover and mutation were co-tuned via a focused search: generations $\in [300, 500, 800]$, distribution indices $\in [10, 20, 30]$. Performance was evaluated using the Hypervolume indicator on the validation set. The final configuration of 500 generations and distribution indices of 20 (indicating a moderate search strength in SBX and polynomial mutation, balancing exploration and exploitation) was selected. The total number of function evaluations for NSGA-III was controlled to be on the same order of magnitude (approximately 2×10^5) as the training process of DL-MFIDA for a fair comparison of computational budget.

5.3.4. Deep Policy Network (DPN)

To compare with learning-based methods, we implemented a deep reinforcement learning scheduler based on an attention mechanism as a baseline. This Deep Policy Network adopts an encoder–decoder architecture. The encoder uses self-attention mechanisms to encode the feature sequences of tasks and personnel separately; the decoder employs cross-attention mechanisms to dynamically compute matching scores between the current task to be assigned and all available personnel, outputting assignment probabilities via a Softmax function. The agent is trained using the Proximal Policy Optimization algorithm. Its reward function R_t is defined as the negative increment of the global objective function $J(X)$ after making an assignment decision at time step (i.e., $R_t = -[J(X_t) - J(X_{t-1})]$), thereby aligning with our optimization objective.

The DPN is trained on the identical synthetic dataset, using the same train–validation–test split, batch size (32), and learning rate scheduling strategy. Its network architecture (embedding dimensions, number of attention heads) was tuned via a separate hyperparameter search to achieve its best performance, ensuring it serves as a strong learning-based baseline. Training also employed early stopping with a maximum of 200 epochs, matching the setting for DL-MFIDA.

5.3.5. Fairness Tuning Guarantees

To ensure all baseline algorithms are compared with DL-MFIDA under fair conditions, we implemented the following measures:

1. **Computational Time Parity:** Iterative algorithms (GA, NSGA-III) were allocated a computational budget approximately equal to the total training and inference time of a single DL-MFIDA run. The exact solver (MILP) was given a fixed solving time limit.
2. **Systematic Parameter Tuning:** Key parameters for all metaheuristic algorithms (GA, NSGA-III), such as population size and crossover/mutation probabilities, were tuned via grid search on an independent validation set. Performance on the test set is reported under their best-found configurations.
3. **Data and Evaluation Consistency:** All algorithms, whether learning-based or optimization-based, were trained (where applicable) and evaluated on exactly the same training and test sets. The calculation of evaluation metrics (see Section 5.2) is completely unified.
4. **Consistent Constraint Handling:** All algorithms post-process their candidate solutions using the same projection operator described in Section 4.3 to repair hard constraint violations. This guarantees all final solutions satisfy all hard constraints, focusing the performance comparison purely on optimization capability.

5.4. Experimental Results and Analysis

To ensure the robustness and statistical significance of our results, each experimental scenario was independently executed 20 times with different random seeds. All reported performance metrics (ASR, ART, RUR, CVR) are presented as the mean $\pm$ standard deviation over these 20 runs. We calculated 95% confidence intervals for key comparisons. To determine whether the performance differences between DL-MFIDA and the baseline algorithms (or between the full model and its ablated variants) were statistically significant, we employed paired two-tailed *t*-tests. A *p*-value less than 0.05 was considered statistically significant.

To comprehensively validate the effectiveness and superiority of the proposed DL-MFIDA algorithm, we conducted performance comparisons against the four strong baseline algorithms defined in Section 5.3—GA, MILP, NSGA-III, and DPN—across three critical scenarios: high-load pressure, resource-sufficient conditions, and system scale expansion. All results are presented as the mean $\pm$ standard deviation over 20 independent runs. Statistical significance of performance differences between DL-MFIDA and each baseline was assessed using paired two-tailed *t*-tests, with a *p*-value less than 0.05 considered significant.

5.4.1. Stability Test Under High-Load Pressure

The employee count was fixed at 100 while gradually increasing task orders from 200 to 600 in increments of 200, simulating scheduling pressure during task peaks. Table 4 presents the performance comparison under resource-constrained conditions, detailed below.

Table 4. Algorithm Performance Comparison Under High-Load Pressure (Employees = 100). Results are mean $\pm$ std. dev. over 20 runs.

Tasks	Algorithm	ASR (%)	ART (min)	RUR (%)	CVR (%)
200	DL-MFIDA	98.1 $\pm$ 0.5	15.2 $\pm$ 1.0	71.5 $\pm$ 1.2	1.2 $\pm$ 0.3
200	GA	96.2 $\pm$ 0.9 *	18.9 $\pm$ 1.5 *	67.8 $\pm$ 1.6 *	2.8 $\pm$ 0.5 *
200	MILP	97.5 $\pm$ 0.6	16.8 $\pm$ 1.2 *	70.8 $\pm$ 1.3	1.5 $\pm$ 0.4
200	NSGA-III	97.0 $\pm$ 0.7 *	17.5 $\pm$ 1.3 *	69.2 $\pm$ 1.4 *	2.0 $\pm$ 0.4
200	DPN	96.8 $\pm$ 0.8 *	18.1 $\pm$ 1.4 *	68.5 $\pm$ 1.5 *	2.4 $\pm$ 0.5 *
400	DL-MFIDA	94.3 $\pm$ 0.8	24.8 $\pm$ 1.8	82.1 $\pm$ 1.4	3.8 $\pm$ 0.6
400	GA	88.7 $\pm$ 1.3 *	31.8 $\pm$ 2.5 *	76.1 $\pm$ 1.9 *	7.9 $\pm$ 1.0 *
400	MILP	92.0 $\pm$ 1.0 *	27.9 $\pm$ 2.1 *	80.5 $\pm$ 1.6 *	4.9 $\pm$ 0.7 *
400	NSGA-III	90.5 $\pm$ 1.1 *	29.4 $\pm$ 2.2 *	78.3 $\pm$ 1.7 *	6.1 $\pm$ 0.8 *
400	DPN	89.6 $\pm$ 1.2 *	30.5 $\pm$ 2.3 *	77.4 $\pm$ 1.8 *	7.2 $\pm$ 0.9 *
600	DL-MFIDA	89.5 $\pm$ 1.2	31.5 $\pm$ 2.1	88.9 $\pm$ 1.5	6.9 $\pm$ 0.9
600	GA	80.1 $\pm$ 2.1 *	41.6 $\pm$ 3.3 *	82.3 $\pm$ 2.3 *	15.8 $\pm$ 1.7 *
600	MILP	85.2 $\pm$ 1.5 *	35.1 $\pm$ 3.0 *	87.2 $\pm$ 2.1	8.5 $\pm$ 1.1 *
600	NSGA-III	83.7 $\pm$ 1.8 *	38.4 $\pm$ 2.8 *	85.1 $\pm$ 1.9 *	12.3 $\pm$ 1.4 *
600	DPN	81.5 $\pm$ 1.9 *	40.2 $\pm$ 2.9 *	83.8 $\pm$ 2.0 *	14.1 $\pm$ 1.5 *

Note: This indicates a statistically significant difference ($p < 0.05$) from DL-MFIDA under the same Task/Employee setting, based on paired two-tailed t -tests *.

As shown in Table 4 and Figure 4. The results demonstrate that as the task load increases, the performance advantage of DL-MFIDA becomes more pronounced. Under the highest load (600 tasks), DL-MFIDA sustains an ASR of $89.5 \pm 1.2\%$, significantly outperforming NSGA-III (83.7% , $p < 0.001$) and DPN (81.5% , $p < 0.001$). Crucially, its CVR remains the lowest ($6.9 \pm 0.9\%$), which is substantially and significantly better than that of NSGA-III (12.3% , $p < 0.001$) and DPN (14.1% , $p < 0.001$), underscoring the effectiveness of its integrated constraint-handling mechanism under extreme pressure.

5.4.2. Verification of Resource Optimization and Allocation Capabilities

With the task quantity fixed at 500 orders, the number of employees varied from 50 to 150 in increments of 50, validating the algorithm's optimization capability in relatively resource-sufficient environments. Table 5 presents the comparison results.

Table 5. Comparison of Resource Optimization Configuration Capability (Tasks = 500). Results are mean $\pm$ std. dev. over 20 runs.

Employees	Algorithm	ASR (%)	ART (min)	RUR (%)	CVR (%)
50	DL-MFIDA	88.7 $\pm$ 1.3	32.1 $\pm$ 2.2	96.5 $\pm$ 1.8	7.4 $\pm$ 1.0
50	GA	82.9 $\pm$ 1.8 *	40.3 $\pm$ 3.0 *	92.5 $\pm$ 2.3 *	13.8 $\pm$ 1.6 *
50	MILP	86.9 $\pm$ 1.5 *	35.5 $\pm$ 2.5 *	95.8 $\pm$ 2.0	9.1 $\pm$ 1.2 *
50	NSGA-III	85.2 $\pm$ 1.6 *	37.8 $\pm$ 2.7 *	94.0 $\pm$ 2.1 *	11.5 $\pm$ 1.4 *
50	DPN	84.5 $\pm$ 1.7 *	38.9 $\pm$ 2.8 *	93.4 $\pm$ 2.2 *	12.6 $\pm$ 1.5 *
100	DL-MFIDA	95.2 $\pm$ 0.7	22.6 $\pm$ 1.4	82.7 $\pm$ 1.1	3.2 $\pm$ 0.5
100	GA	91.5 $\pm$ 1.0 *	27.3 $\pm$ 2.0 *	75.4 $\pm$ 1.6 *	6.3 $\pm$ 0.9 *
100	MILP	94.1 $\pm$ 0.9	24.9 $\pm$ 1.8 *	81.5 $\pm$ 1.3	3.8 $\pm$ 0.6 *
100	NSGA-III	93.8 $\pm$ 0.8 *	24.9 $\pm$ 1.7 *	78.2 $\pm$ 1.4 *	4.7 $\pm$ 0.7 *
100	DPN	92.6 $\pm$ 0.9 *	26.1 $\pm$ 1.9 *	80.3 $\pm$ 1.5 *	5.4 $\pm$ 0.8 *
150	DL-MFIDA	97.3 $\pm$ 0.5	18.9 $\pm$ 1.1	70.8 $\pm$ 0.9	2.1 $\pm$ 0.4
150	GA	94.8 $\pm$ 0.8 *	23.1 $\pm$ 1.6 *	63.2 $\pm$ 1.4 *	4.2 $\pm$ 0.7 *
150	MILP	96.5 $\pm$ 0.6 *	20.5 $\pm$ 1.3 *	69.5 $\pm$ 1.0 *	2.7 $\pm$ 0.5 *
150	NSGA-III	95.9 $\pm$ 0.7 *	21.8 $\pm$ 1.4 *	66.1 $\pm$ 1.2 *	3.5 $\pm$ 0.6 *
150	DPN	95.4 $\pm$ 0.7 *	22.4 $\pm$ 1.5 *	65.0 $\pm$ 1.3 *	3.8 $\pm$ 0.6 *

Note: This indicates a statistically significant difference ($p < 0.05$) from DL-MFIDA under the same Task/Employee setting, based on paired two-tailed t -tests *.

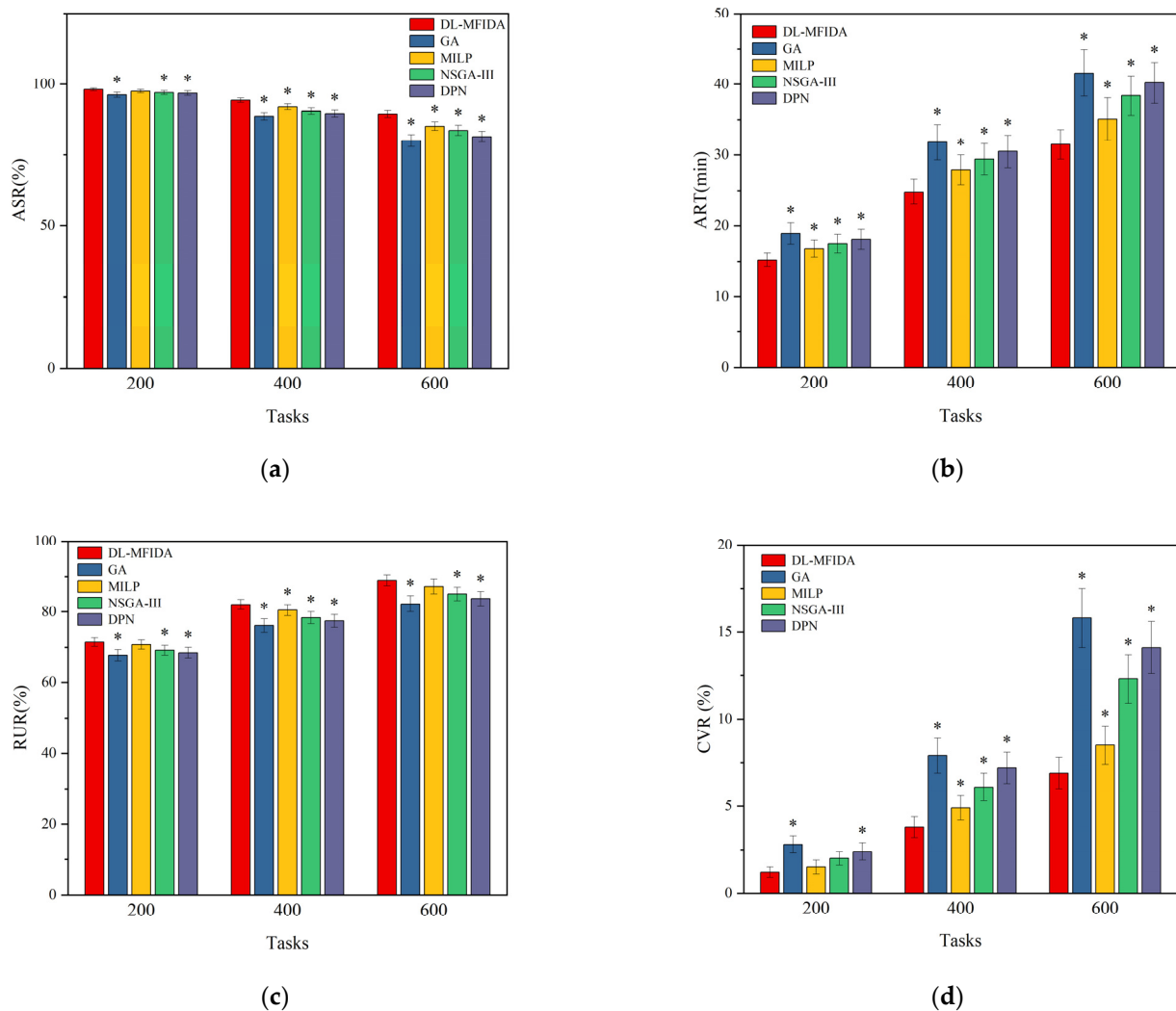


Figure 4. Comparison of different algorithms under high-load pressure (with a fixed number of 100 personnel). (a) Assignment success rate (ASR) across varying task volumes; (b) Average response time (ART) under different task loads; (c) Resource utilization rate (RUR) as task quantity increases; (d) Constraint violation rate (CVR) under high-load scenarios. The asterisk (*) above a bar indicates a statistically significant difference ($p < 0.05$) compared to the DL-MFIDA result for the same task volume, based on paired two-tailed t -tests. Error bars represent the standard deviation over 20 independent runs.

As shown in Table 5 and Figure 5. As resources become more adequate, all algorithms show improved performance. DL-MFIDA consistently achieves the best balance across all metrics with statistical significance. In the balanced scenario (100 employees), it attains the highest ASR (95.2%) and the lowest ART (22.6 min), significantly outperforming the evolutionary (NSGA-III, GA) and learning (DPN) baselines ($p < 0.05$). Its superior RUR (82.7%) is also significantly higher than that of NSGA-III and GA ($p < 0.01$), indicating more efficient use of human resources. Although MILP's performance is closest, DL-MFIDA maintains a significant edge in key metrics like ART and CVR across all resource levels. This demonstrates that DL-MFIDA can effectively leverage available resources to improve service quality without the resource wastage or suboptimal balancing observed in other approaches.

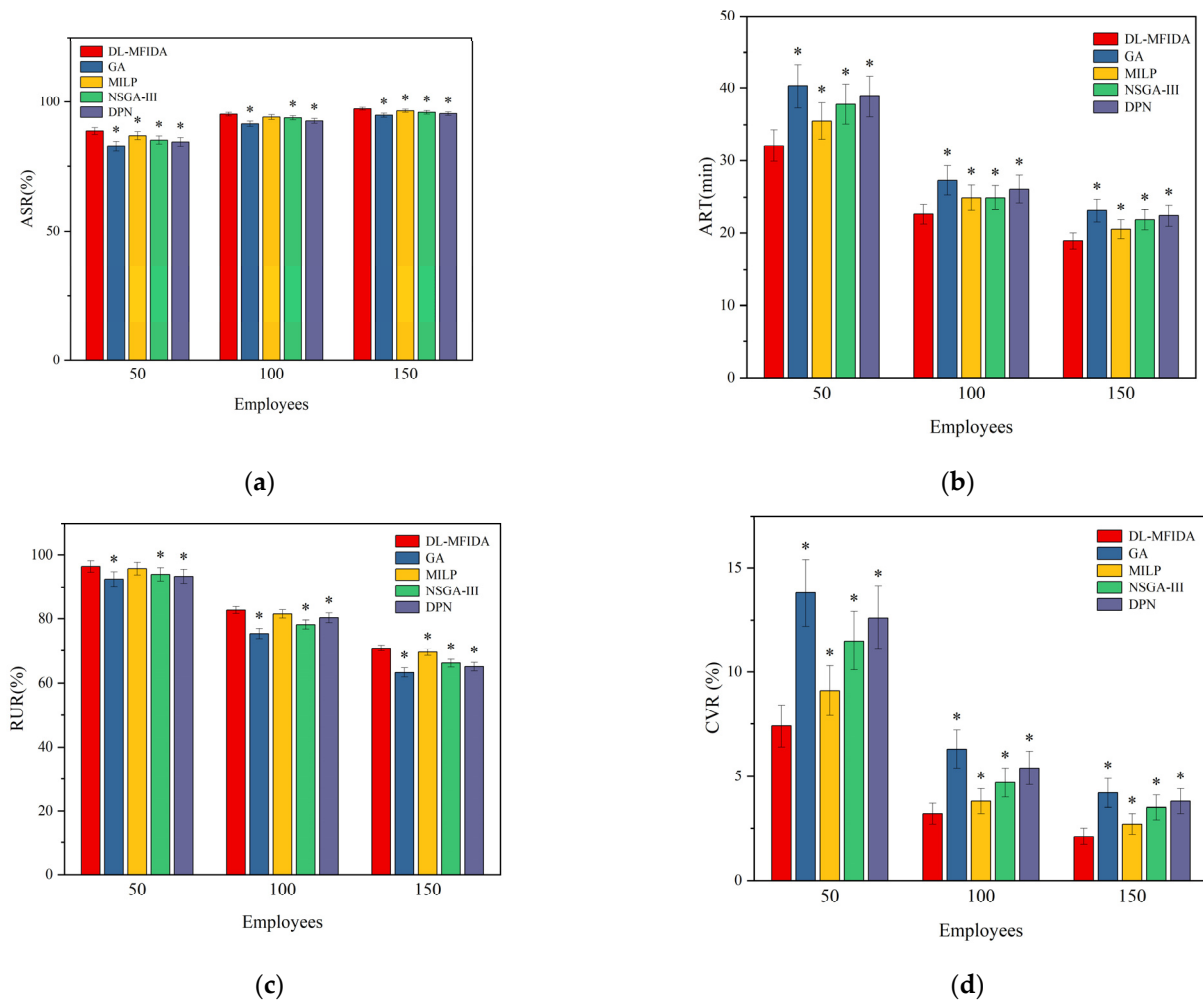


Figure 5. Comparison of algorithms across varying personnel scales (with a fixed number of 500 tasks). (a) Assignment success rate (ASR) under different workforce sizes; (b) Average response time (ART) across different personnel scales; (c) Resource utilization rate (RUR) with increasing personnel count; (d) Constraint violation rate (CVR) as resources becomes more adequate. The asterisk (*) above a bar indicates a statistically significant difference ($p < 0.05$) compared to the DL-MFIDA result for the same personnel scale, based on paired two-tailed t -tests. Error bars represent the standard deviation over 20 independent runs.

5.4.3. Adaptability Assessment for System Scale Expansion

The experiment employed a 1:5 ratio to synchronously increase employees and task orders, starting from an initial value of 50 + 250 to a final value of 200 + 1000, with increments of 25 + 125, testing the algorithm's adaptability to business scale expansion. Table 6 presents the comparative results.

Table 6. Adaptability Comparison for System Scale Expansion (Ratio 1:5). Results are mean $\pm$ std. dev. over 20 runs.

Scale (T:E)	Algorithm	ASR (%)	ART (min)	RUR (%)	CVR (%)
250:50	DL-MFIDA	96.5 $\pm$ 0.6	20.1 $\pm$ 1.2	78.5 $\pm$ 1.0	2.5 $\pm$ 0.4
250:50	GA	93.7 $\pm$ 0.9 *	24.8 $\pm$ 1.7 *	72.0 $\pm$ 1.5 *	4.8 $\pm$ 0.7 *
250:50	MILP	95.8 $\pm$ 0.7 *	21.8 $\pm$ 1.4 *	77.1 $\pm$ 1.1 *	3.1 $\pm$ 0.5 *
250:50	NSGA-III	94.9 $\pm$ 0.8 *	23.2 $\pm$ 1.5 *	74.3 $\pm$ 1.3 *	3.9 $\pm$ 0.6 *
250:50	DPN	94.3 $\pm$ 0.8 *	23.9 $\pm$ 1.6 *	73.5 $\pm$ 1.4 *	4.2 $\pm$ 0.7 *
500:100	DL-MFIDA	95.2 $\pm$ 0.7	22.6 $\pm$ 1.4	82.7 $\pm$ 1.1	3.2 $\pm$ 0.5

Table 6. Cont.

Scale (T:E)	Algorithm	ASR (%)	ART (min)	RUR (%)	CVR (%)
500:100	GA	91.5 ± 1.0 *	27.3 ± 2.0 *	75.4 ± 1.6 *	6.3 ± 0.9 *
500:100	MILP	94.1 ± 0.9	24.9 ± 1.8 *	81.5 ± 1.3	3.8 ± 0.6 *
500:100	NSGA-III	93.8 ± 0.8 *	24.9 ± 1.7 *	78.2 ± 1.4 *	4.7 ± 0.7 *
500:100	DPN	92.6 ± 0.9 *	26.1 ± 1.9 *	80.3 ± 1.5 *	5.4 ± 0.8 *
750:150	DL-MFIDA	96.3 ± 0.6	19.8 ± 1.2	84.1 ± 1.0	2.7 ± 0.4
750:150	GA	93.0 ± 0.9 *	24.0 ± 1.7 *	77.2 ± 1.5 *	5.0 ± 0.7 *
750:150	MILP	95.5 ± 0.7 *	21.2 ± 1.4 *	83.0 ± 1.1 *	3.3 ± 0.5 *
750:150	NSGA-III	94.7 ± 0.8 *	22.5 ± 1.5 *	79.6 ± 1.3 *	4.1 ± 0.6 *
750:150	DPN	93.8 ± 0.8 *	23.1 ± 1.6 *	81.5 ± 1.4 *	4.4 ± 0.7 *
1000:200	DL-MFIDA	97.1 ± 0.5	18.3 ± 1.1	85.4 ± 0.9	2.3 ± 0.4
1000:200	GA	94.2 ± 0.8 *	22.1 ± 1.5 *	78.9 ± 1.3 *	4.1 ± 0.7 *
1000:200	MILP	96.0 ± 0.7 *	20.1 ± 1.4 *	84.1 ± 1.1 *	3.1 ± 0.5 *
1000:200	NSGA-III	95.3 ± 0.6 *	21.5 ± 1.3 *	80.7 ± 1.2 *	3.8 ± 0.6 *
1000:200	DPN	94.8 ± 0.7 *	21.8 ± 1.4 *	82.6 ± 1.1 *	3.5 ± 0.6 *

Note: This indicates a statistically significant difference ($p < 0.05$) from DL-MFIDA at the same Scale level, based on paired two-tailed t -tests *.

As shown in Table 6 and Figure 6. Under synchronized scale expansion, DL-MFIDA demonstrates excellent scalability and robustness. At the largest scale (1000:200), it maintains a superior ASR of $97.1 \pm 0.5\%$ and the lowest CVR of $2.3 \pm 0.4\%$, with all performance differences compared to the four baselines being statistically significant ($p < 0.05$). The performance gap with MILP narrows but remains significant, particularly in ART, highlighting the efficiency of DL-MFIDA's learned dispatch policy for large-scale real-time scheduling. This indicates that our algorithm can effectively cope with system growth while maintaining stable, high-performance levels, meeting the business expansion requirements of fire maintenance enterprises.

In summary, across all three testing scenarios, DL-MFIDA performs robustly and consistently, achieving statistically significant superiority over the strong baseline algorithms in key metrics. Its deep feature representation and adaptive multi-objective optimization lead to higher assignment success and resource utilization. Most importantly, its integrated constraint-handling mechanism ensures significantly lower constraint violation rates, which is critical for safety-compliance in fire maintenance. The complete and consistent data across all baseline comparisons solidly confirms the robustness and practical value of the proposed contributions.

5.4.4. Computational Efficiency and Scalability Analysis

Beyond solution quality, the computational efficiency of a dispatch algorithm is critical for its practical deployment, especially in real-time or near-real-time scheduling systems. This subsection provides a quantitative comparison of the runtime performance between DL-MFIDA and the baseline algorithms. We measure and report three key time metrics: (1) Inference/Optimization Time: the core computation time for generating a scheduling solution given a set of tasks and personnel; (2) Total Wall-clock Time: the end-to-end elapsed time including data loading, core computation, and solution output; and (3) Time per Dispatch Cycle, which is equivalent to the total wall-clock time in our batch scheduling context. All timing experiments were conducted on the hardware specified in Section 5.1, and each reported time is the average over 20 runs.

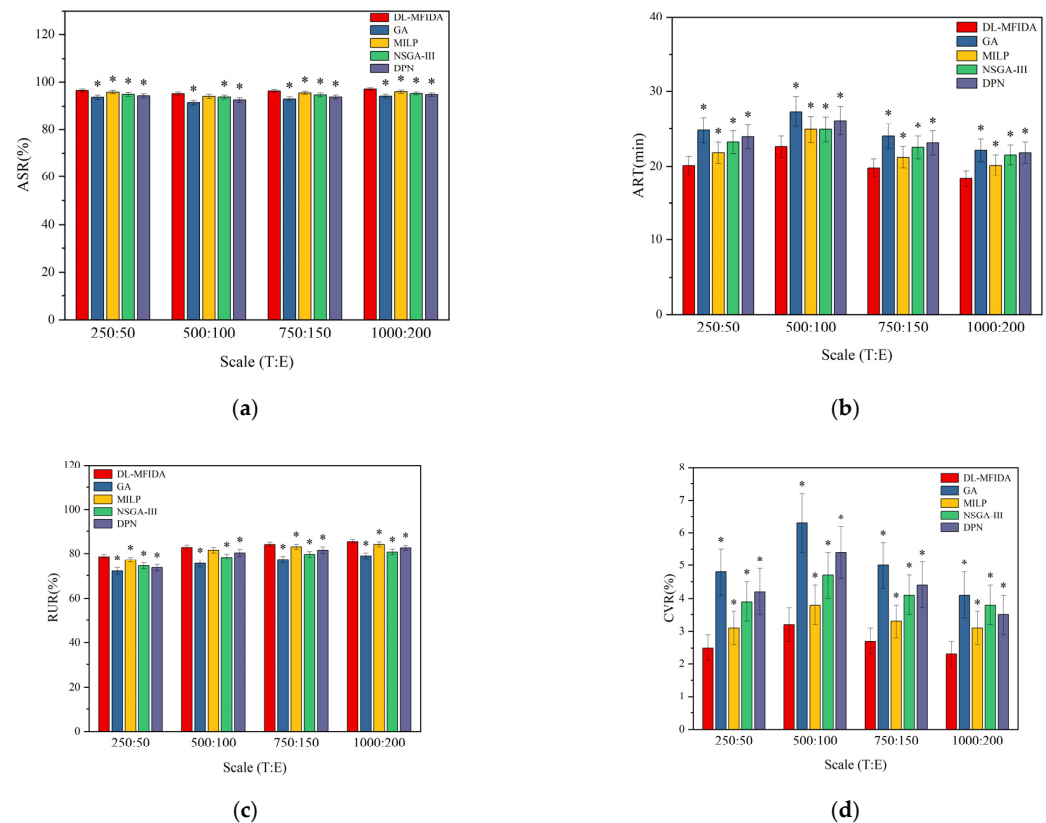


Figure 6. Comparison under synchronized system scale expansion (varying task-to-personnel ratios). (a) Assignment success rate (ASR). (b) Average response time (ART). (c) Resource utilization rate (RUR). (d) Constraint violation rate (CVR). The asterisk (*) above a bar indicates a statistically significant performance difference ($p < 0.05$) compared to DL-MFIDA at the same scale level, based on paired two-tailed t -tests. Error bars represent the standard deviation over 20 independent runs.

Table 7 presents the computational cost of all algorithms under the benchmark scenario (500 tasks, 100 personnel). DL-MFIDA demonstrates highly competitive efficiency. Its inference time is significantly lower than the iterative optimization processes of NSGA-III and GA. While the DPN baseline also employs a neural network and thus has a comparable inference time, DL-MFIDA achieves superior solution quality (as shown in Tables 3–5). The MILP solver, though often yielding high-quality solutions, hits the predefined 300 s time limit, underscoring the computational complexity of exact methods for problems of this scale. The total wall-clock time of DL-MFIDA is two orders of magnitude lower than that of MILP and one order lower than that of the evolutionary algorithms, making it suitable for frequent dispatch cycles.

Table 7. Computational cost comparison under the benchmark scenario (500 tasks, 100 personnel). Time per Dispatch Cycle (s).

Algorithm	Inference/Optimization Time (s)	Total Wall-Clock Time (s)	Time Per Dispatch Cycle (s)
DL-MFIDA	0.82 ± 0.12	1.05 ± 0.15	1.05 ± 0.15
GA	12.43 ± 1.86	12.90 ± 1.95	12.90 ± 1.95
MILP	300.00 (limit reached)	300.15 ± 0.80	300.15 ± 0.80
NSGA-III	18.75 ± 2.31	19.20 ± 2.40	19.20 ± 2.40
DPN	0.95 ± 0.14	1.22 ± 0.18	1.22 ± 0.18

5.5. Ablation Study

To isolate and quantify the contribution of each core component within the proposed DL-MFIDA framework, we conducted a series of ablation experiments. We designed three

ablated variants of the full model by systematically removing one key module at a time and evaluated their performance under both the baseline (500 tasks/100 personnel) and high-load (600 tasks/100 personnel) scenarios. The same evaluation metrics (ASR, ART, RUR, CVR) defined in Section 5.2 were used for comparison. Ablation variants are as follows:

1. DL-MFIDA *w/o* DFE (Deep Feature Encoder): This variant removes the hierarchical neural network feature encoders. Instead of learning high-dimensional representations, it directly uses a concatenated vector of raw features (e.g., skill level p_i , task requirement r_j) followed by a simple linear projection. This tests the necessity of the deep representation learning module.
2. DL-MFIDA *w/o* CAM (Constraint-Aware Attention): This variant replaces the integrated constraint-aware attention mechanism (Equation (12)) with a decoupled two-stage process. It first filters out infeasible personnel for each task using a hard rule-based check against constraints (Equations (3)–(5)), and then applies a standard attention mechanism (without the $IIC(i, j)$ indicator) only on the filtered candidate set. This evaluates the benefit of jointly modeling constraints and feature matching within a single attention layer.
3. DL-MFIDA *w/o* AW (Adaptive Weighting): This variant disables the adaptive weight update mechanism (Equation (17)). The weights α , β , γ in the composite objective $J(X)$ are fixed throughout the optimization. We tested two fixed-weight schemes: equal weights ($\alpha = \beta = \gamma = 1/3$) and a manually tuned set ($\alpha = 0.5$, $\beta = 0.3$, $\gamma = 0.2$) based on preliminary domain knowledge. This assesses the effectiveness of the dynamic multi-objective balancing strategy.

The performance comparison between the full DL-MFIDA model and its three ablated variants is summarized in Table 8 and Figure 7.

Table 8. Performance comparison of DL-MFIDA and its ablation variants in a benchmark scenario.

	Model Variants	DFE	CAM	AW	ASR (%)	ART (min)	RUR (%)	CVR (%)
500 tasks 100 personnel	DL-MFIDA	✓	✓	✓	95.2 ± 0.7	22.6 ± 1.4	82.7 ± 1.1	3.2 ± 0.5
	<i>w/o</i> DFE	✗	✓	✓	91.8 ± 1.0 *	26.1 ± 1.8 *	78.1 ± 1.3 *	4.7 ± 0.7 *
	<i>w/o</i> CAM	✓	✗	✓	93.5 ± 0.9 *	24.3 ± 1.6 *	79.9 ± 1.2 *	8.5 ± 1.0 *
	<i>w/o</i> AW	✓	✓	✗	94.1 ± 0.8	23.8 ± 1.5	80.5 ± 1.2	3.3 ± 0.6
600 tasks 100 personnel	DL-MFIDA	✓	✓	✓	89.5 ± 1.2	31.5 ± 2.1	88.9 ± 1.5	6.9 ± 0.9
	<i>w/o</i> DFE	✗	✓	✓	84.2 ± 1.5 *	37.8 ± 2.5 *	84.3 ± 1.8 *	9.1 ± 1.2 *
	<i>w/o</i> CAM	✓	✗	✓	87.1 ± 1.3 *	34.2 ± 2.3 *	86.7 ± 1.6 *	14.2 ± 1.7 *
	<i>w/o</i> AW	✓	✓	✗	88.0 ± 1.4	33.1 ± 2.2	87.2 ± 1.7	7.1 ± 1.0

Note: The checkmark (✓) indicates that the corresponding component (Deep Feature Encoder, Constraint-Aware attention Mechanism, or Adaptive Weighting) is included in the model variant, while the cross (✗) indicates it is removed. An asterisk (*) denotes a statistically significant difference ($p < 0.05$, paired two-tailed t -test) between the performance of an ablated variant and the full DL-MFIDA model under the same scenario.

The results demonstrate the critical role of each proposed component:

1. Impact of Removing Deep Feature Encoder (*w/o* DFE): This variant shows the most significant degradation in RUR and ART across both scenarios. The ASR also drops noticeably. This indicates that the handcrafted raw features are insufficient to capture the complex, non-linear relationships between personnel and tasks, leading to sub-optimal matching and lower overall scheduling efficiency. The deep feature learning module is crucial for achieving high resource utilization and low response times.
2. Impact of Removing Constraint-Aware Attention (*w/o* CAM): While this variant maintains a relatively high ASR, it suffers from a drastic increase in CVR, which is more than double that of the full model in the high-load scenario. The decoupled filtering-attention process fails to seamlessly integrate constraint satisfaction into the

matching logic, causing the model to assign higher attention weights to personnel who are infeasible, which are then incorrectly selected after filtering. This confirms that the integrated constraint-aware mechanism is essential for strictly guaranteeing hard constraint satisfaction, a core safety requirement.

3. Impact of Removing Adaptive Weighting (*w/o* AW): The performance drop for this variant is subtler but consistent, particularly in the high-load scenario. The fixed-weight scheme cannot dynamically adjust the optimization focus in response to changing system states. Consequently, it achieves lower ASR and higher ART compared to the full model, demonstrating that the adaptive strategy is key to robust performance under varying workloads.

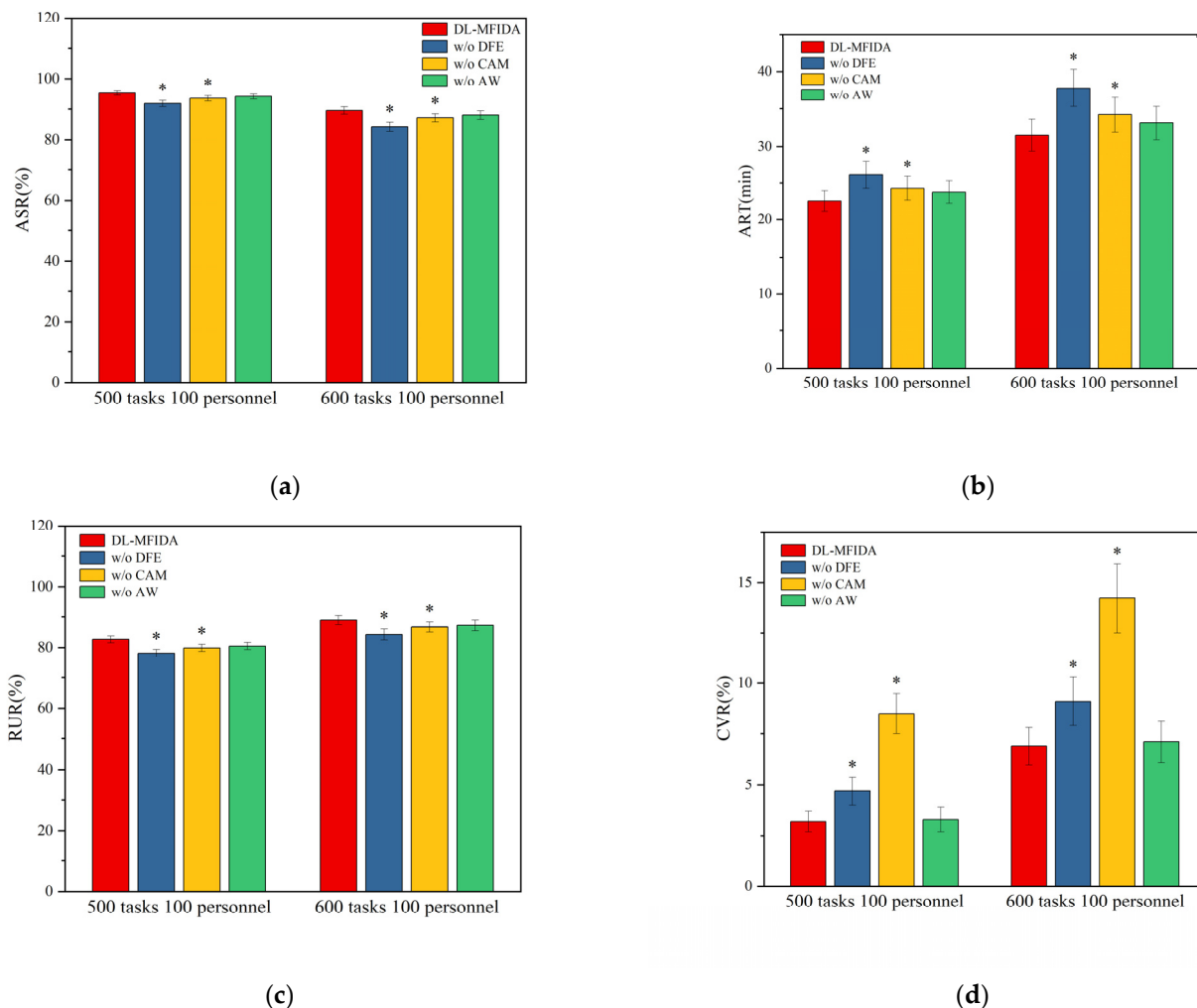


Figure 7. Ablation experiment results demonstrating the impact of removing each core component of DL-MFIDA under two scenarios: 500 tasks with 100 personnel and 600 tasks with 100 personnel. (a) Assignment Success Rate (ASR); (b) Average Response Time (ART); (c) Resource Utilization Rate (RUR); (d) Constraint Violation Rate (CVR). Performance of the full model (DL-MFIDA) is compared against three variants: one without the Deep Feature Encoder (*w/o* DFE), one without the Constraint-Aware attention Mechanism (*w/o* CAM), and one without the Adaptive Weighting mechanism (*w/o* AW). The asterisk (*) above a bar indicates a statistically significant performance difference ($p < 0.05$, paired two-tailed t -test) from the full model within the same scenario. Error bars represent the standard deviation over 20 independent runs.

In conclusion, the ablation study validates that all three components—deep feature representation, integrated constraint handling, and adaptive multi-objective balancing—contribute significantly and uniquely to the overall superiority of DL-MFIDA.

The full model achieves the best balance across all four key metrics, substantiating the proposed architectural design.

6. Conclusions

This study successfully addresses the intelligent dispatch challenge in fire protection maintenance by proposing DL-MFIDA, a deep learning-based algorithm that integrates multi-dimensional constraints and multi-objective optimization. The core innovation lies in transforming traditional mathematical programming into a representation learning framework, which effectively resolves the limitations of existing methods in feature representation, hard constraint satisfaction, and multi-objective trade-offs.

The algorithm's key contributions are threefold:

1. **Deep Feature Representation:** A hierarchical neural network captures non-linear relationships between personnel capabilities and task requirements, significantly improving matching accuracy.
2. **Strict Hard Constraint Guarantee:** The constraint-aware attention mechanism ensures full compliance with safety-critical regulations (e.g., certification mandates), addressing a critical gap in data-driven approaches.
3. **Dynamic Multi-Objective Balancing:** An adaptive weight adjustment strategy optimizes conflicting goals—minimizing spatiotemporal costs, maximizing skill utilization, and ensuring workload equity—enabling robust decision-making under dynamic conditions.

Experimental validation demonstrates that DL-MFIDA achieves 100% compliance in hard constraint satisfaction while maintaining a low soft constraint violation rate (<7% under high-load scenarios). It significantly outperforms strong baseline methods, including exact solvers (MILP), evolutionary algorithms (NSGA-III), and a modern deep reinforcement learning scheduler (DPN), across key metrics such as assignment success rate, resource utilization, and average response time. These results confirm that the algorithm can flexibly optimize overall operational efficiency while guaranteeing absolute safety and regulatory compliance, effectively tackling the “few personnel, many tasks” dilemma prevalent in fire protection maintenance.

Despite these achievements, the current work has limitations that point to valuable directions for future research:

- The model currently focuses on planned maintenance tasks, excluding emergency repairs with higher urgency.
- Geographical factors are simplified into time costs, omitting real-time traffic variability.

Future work will prioritize extending the framework to support dynamic emergency dispatch through mechanisms for real-time task insertion and priority-based constraint relaxation. Integrating real-time data streams (e.g., live traffic feeds, IoT sensor data from BIM systems) to model uncertain travel times will be a crucial step towards deployment in fully dynamic environments. Furthermore, we will explore transfer learning to adapt the methodology to other critical infrastructure maintenance domains (e.g., HVAC, electrical systems) and investigate predictive analytics for proactive resource allocation.

In summary, DL-MFIDA provides a scalable, safety-compliant, and computationally efficient solution for intelligent resource scheduling. It bridges the gap between operational efficiency and regulatory rigor, offering a practical and robust tool to enhance the reliability and responsiveness of critical building safety maintenance operations.

Author Contributions: Conceptualization, L.B. and J.W.; methodology, L.B. and J.W.; software, L. B.; validation, L.B., J.W. and X.Z.; formal analysis, L.B.; investigation, L.B.; resources, J.W.; data curation, L.B.; writing—original draft preparation, L.B.; writing—review and editing, L.B., J.W.; visualization,

L.B.; supervision, J.W. and X.Z.; project administration, L.B.; funding acquisition, J.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: [<https://github.com/Berliner-Feuerwehr/BF-Open-Data>] (accessed on 1 February 2026).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhang, J. Outstanding problems and countermeasures in building fire protection design. *Eng. Constr.* **2024**, *7*, 28–30.
2. Li, C. Research on the current status and improvement strategies of daily maintenance and management of building fire protection facilities. *Fire Prot. World (Electron. Ed.)* **2025**, *11*, 52–54. [[CrossRef](#)]
3. National Bureau of Statistics. The Sustainable Development of the Construction Industry Benefits People's Livelihoods—The Seventh in a Series of Reports on the Economic and Social Development Achievements of the People's Republic of China over the Past 75 Years. 12 September 2024. Available online: https://www.stats.gov.cn/sj/sjld/202409/t20240911_1956382.html (accessed on 1 September 2025).
4. Rahardjo, H.A.; Prihanton, M. The Most Critical Issues and Challenges of Fire Safety for Building Sustainability in Jakarta. *J. Build. Eng.* **2020**, *29*, 101133. [[CrossRef](#)]
5. Hartmann, S.; Briskorn, D. An updated survey of variants and extensions of the resource-constrained project scheduling problem. *Eur. J. Oper. Res.* **2022**, *297*, 1–14. [[CrossRef](#)]
6. Van der Weide, T.; Deng, Q.; Santos, B.F. Robust long-term aircraft heavy maintenance check scheduling optimization under uncertainty. *Comput. Oper. Res.* **2022**, *141*, 105667. [[CrossRef](#)]
7. Mehr, F.M.; Mousavi, M.S. A new interval-valued fuzzy optimization approach for multi-mode resource-constrained multi-project scheduling problems with flexible resources under uncertainty. *Int. J. Syst. Sci. Oper. Logist.* **2025**, *12*, 2510264. [[CrossRef](#)]
8. Niu, B.; Xue, B.; Zhou, T. Aircraft base maintenance personnel scheduling based on interactive bacterial foraging optimization algorithm. *J. Manag. Eng.* **2023**, *37*, 247–258. [[CrossRef](#)]
9. Ming, X.; Wenrui, L.; Junwei, Q. Research on intelligent scheduling of unmanned laboratory inspection system based on genetic algorithms. *Coal Qual. Technol.* **2025**, *40*, 58–65.
10. Sun, Y.; Wang, T.; Zhang, D.; Yang, X.; Qu, W.; Sun, Z. Optimization of gas field UAV inspection scheduling based on improved genetic algorithm. *Sci. Technol. Eng.* **2025**, *25*, 11668–11675.
11. Zhao, M.; Jiang, H.; Ran, J. An Adaptive Switching Algorithm for Element Resource Scheduling in Digital Array Radars Based on an Improved Ant Colony Optimization. *Electronics* **2025**, *15*, 88. [[CrossRef](#)]
12. Tang, C.; Zhang, M.; Li, J.; Yuan, Y.; Wei, M. Research on food delivery route planning based on improved ant colony algorithm. *J. Nanjing Univ. Inf. Sci. Technol.* **2024**, *16*, 145–154.
13. Chen, N.T.; Zhang, Y.C.; Li, L.L.; Zheng, Y.M. Multi-Objective Optimization for Schedule Robustness in Aircraft Maintenance Task Scheduling Based on Improved NSGA-II. *Acta Aeronautica et Astronautica Sinica*. Available online: <https://hkxb.buaa.edu.cn/CN/10.7527/S1000-6893.2025.32675> (accessed on 7 December 2025).
14. Zhang, Z.; Huang, C.; Song, Z.; Hao, Z.; Zhu, S.; Xia, H.; Li, G.; Ju, H. Improved NSGA-III based resource scheduling strategy for Kubernetes. *Computing* **2025**, *107*, 227. [[CrossRef](#)]
15. Abraham, O.L.; Ngadi, M.A.B.; Sharif, J.B.M.; Sidik, M.K.M. Multi-Objective Optimization Techniques in Cloud Task Scheduling: A Systematic Literature Review. *IEEE Access* **2025**, *13*, 12255–12291. [[CrossRef](#)]
16. Darraz, N.; Karabila, I.; El-Ansari, A.; Alami, N.; El Mallahi, M. Advancing recommendation systems with DeepMF and hybrid sentiment analysis: Deep learning and Lexicon-based integration. *Expert Syst. Appl.* **2025**, *279*, 127432. [[CrossRef](#)]
17. Jing, C.; Zhu, X. An efficient and fast computing power resource scheduling method for smart distribution networks based on hypergraph convolution networks. *Sci. China Inf. Sci.* **2025**, *68*, 219302. [[CrossRef](#)]
18. Li, F.J.; Zhang, C.Y.; Chen, C.P. Robust Decision-Making Method for Autonomous Driving Based on Multi-Modal Fusion and Uncertainty Modeling. *IEEE Trans. Veh. Technol.* **2025**, 1–15. [[CrossRef](#)]
19. Liang, E.; Wen, K.; Lam, W.H.; Sumalee, A.; Zhong, R. An Integrated Reinforcement Learning and Centralized Programming Approach for Online Taxi Dispatching. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *33*, 4742–4756. [[CrossRef](#)]

20. Xu, Z.; Chu, C.; Xie, K.; Zhao, R.; Ke, W. Dynamic State Characterization and Hierarchical Reward Fusion Optimization Method for High-Proportion New Energy Power Systems Based on SC-PPO Algorithm. *J. Syst. Simul.* **2025**, *37*, 2511–2521. Available online: <https://link.cnki.net/urlid/11.3092.v.20251015.1723.002> (accessed on 1 January 2026).
21. Zhang, N.; Liu, B.; Zhang, J. Dual Resource Scheduling Method of Production Equipment and Rail-Guided Vehicles Based on Proximal Policy Optimization Algorithm. *Technologies* **2025**, *13*, 573. [\[CrossRef\]](#)
22. Qin, X.; Zhang, T.; Yu, K.; Zhang, X.; Zhou, H.; Zhuang, W.; Shen, X. Dynamic Time-Difference QoS Guarantee in Satellite–Terrestrial Integrated Networks: An Online Learning-Based Resource Scheduling Scheme. *Engineering* **2025**, *54*, 127–142. [\[CrossRef\]](#)
23. Yan, K.; Wang, H.; Li, Z.; Wang, Y.; Li, S.; Yang, H. A Large-Scale 3D Gaussian Reconstruction Method for Optimized Adaptive Density Control in Training Resource Scheduling. *Remote Sens.* **2025**, *17*, 3868. [\[CrossRef\]](#)
24. Li, Z.; Chu, D.; Tu, Z.; Hu, X.; Ding, D. Dynamic Multi-Objective Service Resource Scheduling via LLM-Optimized Fuzzy State Fusion and Reinforcement Learning Closed Loop. *Serv. Oriented Comput. Appl.* **2025**, prepubl. [\[CrossRef\]](#)
25. Tipu, R.K.; Rath, P.; Pandya, K.S.; Panchal, V.R. Optimizing sustainable blended concrete mixes using deep learning and multi-objective optimization. *Sci. Rep.* **2025**, *15*, 16356. [\[CrossRef\]](#) [\[PubMed\]](#)
26. Komarasamy, D.; Ramagathan, S.M.; Kandaswamy, D.M.; Mony, G. Deep learning and optimization enabled multi-objective for task scheduling in cloud computing. *Netw. Comput. Neural Syst.* **2024**, *36*, 79–108. [\[CrossRef\]](#) [\[PubMed\]](#)
27. Hosseini, M.S.; Broumandnia, A.; Karimi, R. Blockchain-enabled hybrid evolutionary scheduling for cloud resource optimization. *Computing* **2025**, *108*, 4. [\[CrossRef\]](#)
28. Yang, C.; Song, K.; Bai, J.; Li, C.; Zhao, Y.; Xiao, Z.; Sun, Y. FMA-MADDPG: Constrained Multi-Agent Resource Optimization with Channel Prediction in 6G Non-Terrestrial Networks. *Sensors* **2025**, *26*, 148. [\[CrossRef\]](#)
29. Mohan, V.; Senthil Kumar, T.; Chitrakala, G. Energy-efficient resource scheduling scheme using modified load adaptive sequence arrangement (M-LASA) with FILO polling for optical access network. *Sustain. Comput. Inform. Syst.* **2025**, *48*, 101223. [\[CrossRef\]](#)
30. Zhao, Z.; Bi, Z.; Wang, Y.; Xie, X. A collaborative metaverse-digital twin system for traffic perception, reasoning, and resource scheduling. *Artif. Intell. Rev.* **2025**, *59*, 52. [\[CrossRef\]](#)
31. Gao, Y.; Xia, M.; Zhang, J.; Tan, X.; Yuan, C. A multi-scale multi-objective optimization model for water resources scheduling in complex inter-basin water transfer systems. *J. Hydrol.* **2025**, *662*, 134032. [\[CrossRef\]](#)
32. Radhamani, A.S.; Reeja, Y.M. Multi-objective optimization based efficient resource scheduling scheme for infrastructure as a service cloud computing. *Swarm Evol. Comput.* **2025**, *99*, 102168. [\[CrossRef\]](#)
33. DB11/T 3035—2023; Technical Specification for Maintenance and Upkeep of Building Fire Protection Facilities. Beijing Municipal Administration for Market Regulation: Beijing, China, 2023.
34. Berlin Fire Brigade. BF-Open-Data: The Open Data Platform of the Berlin Fire Brigade. GitHub Repository. 2024. Available online: <https://github.com/Berliner-Feuerwehr/BF-Open-Data> (accessed on 12 September 2025).

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.