Multi-Task Deep Reinforcement Learning for Intelligent Logistics Path Planning and Scheduling Optimization

Xianfeng Zhu
School of Economics and Management, Jiaozuo University, Jiaozuo 454000, Henan, China
E-mail: zhu_xianfeng@outlook.com

Keywords: deep reinforcement learning, multi-task learning, logistics optimization, path planning, dynamic demand

Received: January 8, 2025

Intelligent logistics systems optimize path planning and scheduling problems by introducing deep reinforcement learning (DRL) to cope with the complexity of dynamic demand and resource constraints. This study proposes an improved strategy that combines multi-task learning (MTL) with Q-learning to achieve simultaneous optimization of multiple related subtasks, such as path selection, task scheduling, and resource allocation, and shares some network parameters to improve model efficiency and generalization ability. The experimental design covers a variety of logistics scenarios, including urban distribution, long-distance transportation, and emergency response. Five baseline models are used for comparative evaluation to verify the advantages of the DRL method in computational efficiency, cost control, resource utilization, service level, and dynamic adaptability. Through experimental verification, the DRL model achieved a 15% cost reduction in cost control compared to traditional algorithms; the resource utilization rate reached 85%, and it performed excellently in terms of efficiency improvement. It has excellent adaptability when dealing with dynamic demands and can respond quickly to environmental changes, effectively improving the overall performance of the intelligent logistics system. It is particularly suitable for application scenarios that require real-time decision-making and support dynamic demand fluctuations.

Povzetek: Opisana je optimizacijo poti in urnika v pametnih logističnih sistemih z uporabo globokega ojačevalnega učenja (DRL) in večnalognega učenja za učinkovito prilagajanje na dinamične zahteve, izboljšanje izkoriščenosti virov in zmanjšanje stroškov.

1 Introduction

With the rapid development of the global economy and the popularization of e-commerce, the logistics industry has become a core link in the modern supply chain. The concept of intelligent logistics has emerged. It combines technologies such as the Internet of Things, big data, and artificial intelligence to significantly improve the efficiency and flexibility of the logistics system by dynamically optimizing resource allocation and process management. In this context, logistics path planning and scheduling optimization have gradually become key areas for improving logistics efficiency. Whether it is intra-city distribution, long-distance transportation, or international logistics, these fields are faced with the common challenge of how to complete cargo transportation at the lowest cost and fastest speed [1].

In traditional logistics path planning, commonly used algorithms include Dijkstra algorithm and A algorithm, which can find the shortest path in a static environment. However, actual logistics scenarios are full of dynamic changes, such as fluctuations in order quantity, changes in traffic conditions, and sudden resource constraints. This makes it difficult for traditional algorithms to effectively cope with complex dynamic environments [2]. In addition, logistics scheduling, as a supplementary link to path planning, also needs to optimize the matching and allocation of vehicles and goods in real time under multiple resources and multiple constraints. How to solve path planning and scheduling problems at the same time from a global perspective has become an important direction for the development of intelligent logistics [3].

In recent years, deep reinforcement learning (DRL), as an important technology in the field of artificial intelligence, has shown great potential in solving complex decision-making problems. By combining deep learning and reinforcement learning, DRL can learn optimal strategies in high-dimensional state spaces, providing a new approach for dynamic logistics optimization. In particular, in uncertain environments, DRL can dynamically adjust strategies based on real-time feedback, giving it significant advantages in path planning and scheduling optimization [4].

The core goal of logistics path planning and scheduling optimization is to meet user needs at the lowest cost and achieve global optimization under multiple constraints. However, traditional optimization methods usually work in a static or semi-dynamic manner and lack the ability to deeply adapt to real-time and dynamic conditions. This results in limited optimization effects of the system when facing a complex logistics environment. Specifically, the dynamics in the logistics system include real-time changes in order demand, updates to vehicle status, fluctuations in traffic conditions, etc. These factors

put forward higher requirements for path planning and scheduling [5].

This study aims to explore a logistics path planning and scheduling optimization method based on deep reinforcement learning, focusing on solving the following problems: How to quickly respond to external changes in a dynamic environment and adjust vehicle paths and scheduling strategies in real time? How to balance global optimization and local real-time performance in complex scenarios to improve the overall efficiency and service quality of the system? By designing a deep reinforcement learning model for intelligent logistics, the study will further optimize the performance of the algorithm and provide feasible suggestions for industry applications [6,7].

This paper focuses on logistics path planning and real-time scheduling optimization based on deep reinforcement learning. The content structure is as follows: The second part will review related research, including traditional path planning and scheduling optimization methods, the basic principles of deep reinforcement learning, and its current application status in logistics optimization. The third part introduces problem modeling, defines the path planning and scheduling optimization problems in logistics systems in detail, and clarifies the objective function and constraints of the research. The fourth part proposes algorithm design based on deep reinforcement learning, including environment construction, algorithm selection and improvement, and key technologies in the training process. The fifth part is experimental design and analysis, which verifies the performance of the proposed method through comparative experiments and discusses its applicability and scalability in different scenarios. The sixth part explores actual application scenarios and analyzes the integration scheme and implementation effect of the algorithm in the intelligent logistics system. Finally, the seventh part summarizes the research results, points out the existing deficiencies, and looks forward to possible research directions in the future.

The goal of this study is to explore an efficient and intelligent logistics path planning and scheduling optimization method based on deep reinforcement learning to cope with complex and changing logistics environments. The specific research questions are as follows: How to make the intelligent logistics system respond quickly to external changes in a dynamic environment and adjust vehicle paths and scheduling strategies in real time? In complex scenarios, how to balance global optimization and local real-time performance to improve the overall efficiency and service quality of the system? In particular, in terms of dynamic adaptability, how to improve the system's adaptability to real-time changes in order demand and dynamic adjustments in traffic conditions; in terms of global optimization goals, how to achieve the comprehensive goals of minimum logistics cost, optimal resource utilization and highest service level under various constraints.

This study will use five baseline models, namely Dijkstra algorithm, A algorithm, genetic algorithm, simulated annealing algorithm and priority scheduling algorithm, to conduct a comprehensive comparative evaluation with the proposed deep reinforcement learning (DRL) based method to verify the advantages of DRL method in intelligent logistics path planning and scheduling optimization. The DRL model proposed in this study is specially designed for the characteristics of dynamic demand fluctuations in logistics systems, and can effectively cope with real-time dynamic conditions and achieve efficient path planning and scheduling optimization.

Logistics model assumptions and relevance to real-world scenarios:

In the logistics model, it is assumed that the vehicle's cargo capacity is limited, for example, the maximum cargo capacity of each transport vehicle is 10 tons, which is consistent with the actual cargo capacity limit of vehicles in reality, ensuring the feasibility of the model in actual logistics transportation. The order frequency is assumed to be that the number of orders will increase significantly during peak hours (such as 9 to 11 a.m. and 3 to 5 p.m. on weekdays), with the number of orders reaching about 200 orders per hour during peak hours, and relatively reduced during off-peak hours, with the number of orders per hour being about 50 orders during off-peak hours. This assumption is consistent with the law of order demand changing over time in actual logistics. The storage capacity of the warehouse is also set to a limited value, such as the maximum storage capacity of the warehouse is 5,000 cubic meters, reflecting the limitations of warehouse space in reality. These assumptions enable the model to simulate real-life logistics scenarios more realistically and improve the practicality of the research results.

2 Literature review

In recent years, with the rapid development of e-commerce and the complexity of global supply chains, intelligent logistics has become a key means to improve logistics efficiency and service quality. In intelligent logistics systems, path planning and scheduling optimization are core links that directly affect the efficiency and cost of logistics operations. Traditional path planning algorithms, such as the Dijkstra algorithm and the A algorithm [8,9], are mainly used in static environments and are difficult to cope with dynamic changes in actual logistics scenarios. To this end, researchers have proposed a variety of dynamic scheduling and path optimization methods to adapt to complex and changing logistics needs. In recent years, deep reinforcement learning (DRL), as a cutting-edge technology in the field of artificial intelligence, has demonstrated its application potential in logistics optimization. This paper will review the current research status of intelligent logistics path planning and scheduling optimization in combination with specific literature, focusing on the application of deep reinforcement learning in this field, and exploring the deficiencies and improvement directions of existing research [10,11].

2.1 Current status of research on intelligent logistics and path planning

Intelligent logistics systems achieve automation and intelligence in the logistics process by integrating information technology, automation equipment, and optimization algorithms. In terms of path planning, traditional static algorithms such as the Dijkstra algorithm and the A algorithm are widely used to determine the shortest path. However, these algorithms assume that the environment is static and cannot adapt to dynamic changes in actual logistics scenarios, such as traffic congestion and road closures. To this end, researchers have proposed a variety of dynamic scheduling and path optimization methods to cope with complex and changing logistics needs. For example, for the intelligent scheduling and path planning of warehouse logistics robot clusters, researchers have established a flexible and reconfigurable warehouse space model, formulated the operating rules of mobile robot clusters, and conducted method research and simulation verification on intelligent scheduling and path planning, indicating that automated warehousing technology based on mobile robot clusters is expected to play a key role in e-commerce logistics [12].

2.2 Application of deep reinforcement learning in logistics optimization

Deep reinforcement learning (DRL) combines the advantages of deep learning and reinforcement learning and is able to learn optimal strategies in high-dimensional state spaces. In the field of logistics optimization, DRL is used to solve complex path planning and scheduling problems. For example, researchers have proposed a dynamic multi-model green vehicle path optimization method based on deep reinforcement learning, aiming to reduce carbon emissions during transportation and achieve energy conservation, emission reduction and green transportation [13,14]. In addition, DRL has also been used in the field of resource optimization, such as resource balancing, resource allocation and packing problems, showing its broad application prospects in logistics and supply chain management.

2.3 Deficiencies of the current research and directions for improvement

Although deep reinforcement learning has shown great potential in logistics optimization, there are still some challenges and shortcomings. First, the training process of the DRL algorithm in complex logistics scenarios may require a lot of computing resources and time, affecting its practical application. Second, the dynamic and uncertain nature of the logistics environment increases the difficulty of model training, which may lead to insufficient generalization of the model. In addition, existing research focuses on the optimization of a single link and lacks comprehensive optimization considerations for the overall logistics system. To this end, future research can consider the following improvement directions: 1) Develop a more efficient DRL algorithm to reduce training time and computing resource consumption; 2) Enhance the model's adaptability to dynamic environments and improve its generalization ability; 3) Combine multi-agent systems to achieve comprehensive optimization of the overall logistics system.

In summary, intelligent logistics path planning and scheduling optimization are key links to improve logistics efficiency. As an emerging technology, deep reinforcement learning has shown its application potential in this field. However, existing research still has shortcomings. In the future, it is necessary to further explore more efficient and more dynamic DRL algorithms and achieve global optimization of the logistics system to meet the needs of modern logistics.

Table 1: Comparison of baseline methods in logistics path planning and scheduling optimization

Baseline Methods	Accuracy	Cost-Benefit	Multitask Learning Ability	Dynamic Adaptability
Dijkstra's Algorithm	Can accurately calculate the shortest path in a static environment, but the accuracy drops significantly in a dynamic environment	Path selection is relatively fixed, and the cost-benefit is average	Unable to handle multi tasks	Poor adaptability to dynamic changes
A Algorithm	Has certain adaptability in a dynamic environment, but the accuracy is affected by the complexity of the environment	Considers heuristic information, and the cost-benefit is slightly better	Does not possess multi task-handling ability	Inadequate adaptability to complex dynamic changes

Table 1 compares the performance of two common benchmark methods in the field of logistics path planning

and scheduling optimization, Dijkstra algorithm and A algorithm, in several key aspects. Accuracy refers to the accuracy of calculating the optimal path. In static environments, Dijkstra algorithm performs well but performs poorly in dynamic scenarios; A algorithm has certain adaptability in dynamic environments, but its accuracy is affected by the complexity of the environment. Cost-effectiveness reflects the economic benefits of the path selection strategy. Dijkstra algorithm has a fixed path selection pattern, resulting in average cost-effectiveness, while A algorithm shows slightly better cost-effectiveness due to the consideration of heuristic information. Multi-task learning ability refers to the ability to handle multiple related tasks simultaneously. Both algorithms are unable to handle multiple tasks such as integrated path planning, task scheduling, and resource allocation. Dynamic adaptability measures the ability of the algorithm to adjust to changing environmental conditions. Both algorithms have limitations in adapting to dynamic and complex changes in the logistics environment, which is a key factor in actual logistics operations.

3 Problem modeling

In the optimization of intelligent logistics systems, deep reinforcement learning (DRL) has become an important research direction. By combining the advantages of deep learning with reinforcement learning, efficient path planning and scheduling optimization can be achieved in complex logistics environments. To this end, it is necessary to establish a suitable mathematical model to describe the logistics system, demand and resource allocation characteristics, mathematical formulas for path planning and scheduling problems, and clarify the objective function and constraints [15].

3.1 Logistics system description and assumptions

In this study, a logistics system with dynamic demand and timeliness requirements is considered. Assume that the logistics network consists of several nodes and paths, where each node represents a warehouse, distribution center or transportation station, and the path represents the transportation route of goods from one node to another. The logistics network can be represented by a weighted directed graph $G(V, E)$ To indicate that V is a node set, E is a set of edges. Each edge $(u,v) \in E$ Represents the slave node u To Node v The path and edge weights c_{uv} Represents the transportation cost or time consumption of the path [16,17] .

Assume that the operation of the logistics system is subject to the following constraints: the resources in the system include transport vehicles, goods, warehouses, etc. The usage and availability of each resource may change over time; the demand changes over time and may be uncertain. For example, customer demand may fluctuate in different time periods. Based on these assumptions, the goal of the system is to optimize the allocation of resources in the logistics network through reasonable path planning and scheduling, thereby improving transportation efficiency and reducing costs [18].

3.2 Logistics network structure

The logistics network structure usually consists of multiple distribution centers, warehouses, and customer nodes, forming a complex multi-level network. In order to simplify the problem, the following standardized structure can be adopted: distribution center, responsible for receiving and distributing goods, there may be multiple; warehouse node, where goods are stored, transfer station between distribution center and customers; customer node: the final destination of logistics services, demand fluctuates over time. For the path planning problem, the shortest path problem from the starting node to the target node is considered. The choice of path should not only consider the transportation time and cost, but also consider multiple factors such as vehicle capacity and cargo type [19,20].

3.3 Dynamic demand and resource allocation characteristics

In the intelligent logistics system, demand changes over time and is dynamic. Set a time point t When the node i The demand is $d_i(t)$, that is, the demand from customers or distribution centers. These demands may be sudden, periodic or random. The change of dynamic demand requires the logistics system to have a flexible response mechanism to cope with demand fluctuations [21,22].

Resource allocation characteristics are also an important factor in problem modeling. $r_k(t)$ Indicates k The remaining amount of various resources (such as transport vehicles, warehouse space, etc.) needs to be dynamically adjusted according to current demand. The goal of the system is to ensure the optimal use of various resources in the logistics network through reasonable resource scheduling.

In the resource allocation process, a resource scheduling function can be introduced $R(t)$, this function is based on the current time t and demand $d(t)$ Determine the allocation of each resource. In order to maintain the efficient operation of the system, the resource constraints in formula (1) must be met [23,24].

$$r_k(t+1) = r_k(t) - d_k(t) + \Delta r_k(t) \quad (1)$$

$\Delta r_k(t)$ represents the replenishment of resource k at time t

3.4 Mathematical modeling of path planning and scheduling problems

The path planning and scheduling problem is essentially a dynamic optimization problem. The goal of path planning is to minimize the total cost in the logistics system. $x_{ij}(t)$ Represents at time t At this moment, the logistics

network starts from the node i To Node j Does a transport path exist on the $x_{ij}(t)=1$, otherwise $x_{ij}(t)=0$.

In order to describe the path planning and scheduling problem, the following mathematical model can be established. The objective function is to minimize the total cost of the logistics network C , including transportation cost and scheduling cost. The objective function is shown in Formula (2) [25].

$$\sum_{i\in N}\sum_{j\in N}C_{ij}(t)x_{ij}(t)+\sum_{k\in K}\lambda_k(t)r_k(t) \quad (2)$$

Where N is the node set, and K is the resource type set. $C_{ij}(t)$ represents the transportation cost from node i to node j at time t, $\lambda_k(t)$ is the scheduling cost coefficient of resource k at time t, and $r_k(t)$ is the usage of resource k at time t. The objective function not only considers the transportation cost, but also reflects the cost of resource allocation through the term $\sum_{k\in K}\lambda_k(t)r_k(t)$. At the same time, the constraint condition $0\le x_{ij}(t)\le 1$ is added to $x_{ij}(t)$, indicating that the range of path selection variables is reasonable to avoid unreasonable infinite paths. In addition, the vehicle quantity set V is defined in the model, and the vehicle quantity related calculations are associated in subsequent formulas, such as considering the impact of factors such as vehicle carrying capacity on decision-making in resource allocation and path planning.

The constraints include the following points: Demand constraints, the demand of each node must be met as shown in formula (3).

$$\sum_{j=1}^{|V|}x_{ij}(t)=d_i(t),\quad \forall i\in V \quad (3)$$

Resource constraints, The resource usage is as shown in Formula (4) and does not exceed the available amount .

$$r_k(t)\le R_k^{max},\quad \forall k\in\{1,2,\ldots,K\} \quad (4)$$

Path selection constraints: Path selection must satisfy the physical feasibility and time constraints of formula (5).

$$x_{ij}(t)\in\{0,1\},\quad \forall(i,j)\in E \quad (5)$$

3.5 State, action, and reward function

In deep reinforcement learning, the agent learns the optimal strategy by interacting with the environment. After converting the path planning and scheduling problem into a reinforcement learning problem, we first need to define the state of the environment, the actions that the agent can take, and how to evaluate the effect of the action based on the reward.

In time t At this moment, the state of the system $s(t)$ is expressed by Formula 6, including the demand of each node in the current logistics network, the usage of various resources, the choice of transportation path, etc.

$$s(t)=\{d_1(t),d_2(t),\ldots,d_{|V|}(t),r_1(t),r_2(t),\ldots,r_K(t),x_{ij}(t)\} \quad (6)$$

In Status $s(t)$ The actions that the agent can choose are $a(t)$ It means in time t The path and resource scheduling scheme selected at each moment is expressed by Formula (7). For example, i To Node j path, or allocate some resource.

$$a(t)=\{x_{ij}(t)\} \quad (7)$$

The reward function defines the immediate feedback after the agent takes a certain action, which is usually related to factors such as the total cost of the system, transportation efficiency, resource usage, etc. The reward function can be defined as Formula (8).

$$r(t)=-C(t) \quad (8)$$

In Formula (8), $C(t)$ For time t The negative sign indicates that we want to minimize the cost.

$C(t)$ is the total cost of the logistics system at time t , and its calculation method is consistent with the cost calculation in the objective function of Formula 2, that is,

$$C(t)=\sum_{i\in N}\sum_{j\in N}C_{ij}(t)x_{ij}(t)+\sum_{k\in K}\lambda_k(t)r_k(t)$$

By defining appropriate states, actions, and reward functions, deep reinforcement learning algorithms can be used to learn optimal path planning and resource scheduling strategies, thereby optimizing the operation of logistics systems in an environment with dynamic demand and resource constraints.

4 Improved strategies and algorithm optimization

As the application of deep reinforcement learning (DRL) in intelligent logistics path planning and scheduling optimization gradually deepens, traditional Q-learning algorithms have problems such as low efficiency and difficulty in adapting to the needs of multiple tasks when facing complex tasks and dynamic environments. In order to solve these problems, this study optimizes Q-learning in combination with multi-task learning (MTL), so that the algorithm can handle multiple tasks at the same time and share some network parameters, thereby improving the

training efficiency and generalization ability of the model. This chapter will discuss in detail the optimization strategy of Q-learning combined with multi-task learning, including collaborative learning between tasks, the design of shared parameters, and the optimization of multi-objective reward functions.

4.1 Combination of multi-task learning and q-learning

Multi-task learning (MTL) is a method of learning multiple related tasks simultaneously in the same model. By sharing some model parameters, MTL can effectively transmit and share information between different tasks, thereby improving the learning efficiency and generalization ability of the model. In path planning and scheduling optimization, there are multiple interrelated subtasks, such as path selection, task scheduling, resource allocation, etc., which are highly correlated. Therefore, the use of multi-task learning can optimize multiple tasks simultaneously without increasing too much computational overhead and improve the overall performance of the system.

In the framework of Q-learning, each task corresponds to a Q-value function and can share some network structures. Suppose we have M The goal of Q learning is to find the optimal strategy by optimizing the Q value function of each task. Through multi-task learning, the objective functions of all tasks can be jointly optimized in a unified framework.

4.2 Multi-task learning objective function design

In multi-task learning, our goal is to optimize the loss functions of multiple tasks simultaneously. Suppose there are M tasks, and the loss function of each task is L_m, in $m \in \{1, 2, \ldots, M\}$, we weighted sum these loss functions to get the total loss function. Specifically, for the Q-value functions of path planning tasks, scheduling optimization tasks, and resource allocation tasks, we can construct the objective function as shown in Formula (9).

$$L_{\text{MTL}} = \sum_{m=1}^{M} \lambda_m L_m \tag{9}$$

In Formula (9), λ_m For the task m The weight coefficient indicates the importance of each task in the total loss.

The role of the multi-task loss function in formula (9) is to weighted sum the loss functions of the path planning task, scheduling optimization task and resource allocation task to achieve joint optimization of multiple related tasks. The calculation method is as follows: Assuming there are n tasks, the loss function of each task is L_i, and the weight coefficient of task i is α_i, then the total loss function $L = \sum_{i=1}^{n} \alpha_i L_i$. In actual calculations, the loss function L_1 of the path planning task is calculated based on factors such as the accuracy and cost of path selection; the loss function L_2 of the scheduling optimization task is calculated based on the rationality and cost of the scheduling scheme; and the loss function L_3 of the resource allocation task is determined based on the efficiency and rationality of resource utilization. By adjusting the weight coefficient α_i, the relative importance of different tasks in the total loss can be controlled.

4.3 Multi-task Q-learning loss function

For each task, we can use the loss function of Q learning for optimization. In the path planning task, the loss function can be defined as formula (10).

$$L_{\text{path}} = \frac{1}{N} \sum_{i=1}^{N} \left(Q_{\text{path}}(s_i, a_i) - Q^*_{\text{path}}(s_i, a_i) \right)^2 \tag{10}$$

In Formula (10), $Q_{\text{path}}(s_i, a_i)$ is the Q value in the path planning task, $Q^*_{\text{path}}(s_i, a_i)$ is the target Q value, which represents the optimal strategy for path selection.

In the scheduling optimization task, the loss function can be expressed as Formula (11).

$$L_{\text{scheduling}} = \frac{1}{N} \sum_{i=1}^{N} \left(C_{\text{schedule}}(s_i) - C^*_{\text{schedule}}(s_i) \right)^2 \tag{11}$$

In Formula (11), $C_{\text{schedule}}(s_i)$ is the cost function in the scheduling optimization task, $C^*_{\text{schedule}}(s_i)$ is the target cost.

In the resource allocation task, the loss function can be defined as Formula (12).

$$L_{\text{resource}} = \frac{1}{N} \sum_{i=1}^{N} \left(r_i - r^*(s_i) \right)^2 \tag{12}$$

In Formula (12), r_i is the consumption in the resource allocation task, $r^*(s_i)$ is the target resource consumption.

By sharing network parameters, MTL is able to leverage information from other tasks when learning one task, reducing training time and improving the generalization ability of the model.

4.4 Shared network structure design

In multi-task learning, sharing part of the network structure between tasks can effectively improve training efficiency. Specifically, we can design a shared network layer to extract the common features of the tasks, and then perform personalized optimization in the dedicated network layer of each task. Suppose we have a neural network structure in which the first few layers are used to extract shared features, and the following layers are used to handle the specific needs of each task.

For example, in a path planning task, the network's output layer can predict the Q value of the path selection; while in a scheduling optimization task, the network's output layer can predict the scheduling cost. By sharing the network layer, MTL can fully share information between different tasks, thereby improving the learning effect of each task.

4.5 Optimization strategy of multi-task learning combined with Q-learning

In order to further improve the performance of Q-learning, this study proposes the following optimization strategies for combining multi-task learning with Q-learning:

In multi-task learning, the importance of different tasks may change as the training process progresses. Therefore, we introduced a dynamic task weight adjustment strategy. During the training process, the weights of each task are relatively balanced in the initial stage, but as the training progresses, some tasks may have a greater impact on the overall performance of the system, so it is necessary to dynamically adjust the weight coefficient of the task λ_m .

The task weight can be adjusted based on the learning progress of the task. For example, if the learning progress of the path planning task is slow, the weight of the task can be increased to promote its learning. If the scheduling task has converged, the weight of the task can be reduced to allocate more learning resources to other tasks.

In multi-task learning, in addition to the loss function of each task, the relationship between multiple tasks needs to be considered. In order to balance the goals between different tasks, this study introduces a multi-objective reward function. The total reward function can be expressed as Formula (13).

$$r_t = \lambda_{\text{path}} r_{\text{path}}(s_t, a_t) + \lambda_{\text{scheduling}} r_{\text{scheduling}}(s_t, a_t) + \lambda_{\text{resource}} r_{\text{resource}}(s_t, a_t) \tag{13}$$

By adjusting the reward weight coefficient of each task λ_{path} , $\lambda_{\text{scheduling}}$ and $\lambda_{\text{resource}}$, we can control the contribution of different tasks to the total reward and thus achieve a balance between tasks.

In multi - task learning, the importance of different tasks may change as the training process progresses. Therefore, we introduced a dynamic task weight adjustment strategy. During the training process, the weights of each task are relatively balanced in the initial stage, but as the training progresses, some tasks may have a greater impact on the overall performance of the system, so it is necessary to dynamically adjust the weight coefficient of the task. We define "converged" in the context of the scheduling task as follows: when the change in the loss function of the scheduling task between consecutive training steps is less than a pre - defined small value, we consider the scheduling task to have converged. For instance, if this pre - defined small value is set as 0.001, and the change in the loss function between two consecutive steps meets this criterion, then the scheduling task has converged. The task weight can be adjusted based on the learning progress of the task. For example, if the learning progress of the path planning task is slow, the weight of the task can be increased to promote its learning. If the scheduling task has converged, the weight of the task can be reduced to allocate more learning resources to other tasks.

4.6 Multi-task learning strategy optimization based on Q-learning

In order to improve the combination of multi-task learning and Q-learning, this study introduced the Prioritized Experience Replay (PER) strategy. Experience Replay is a technique used in reinforcement learning to alleviate the correlation between data. It can break the temporal correlation between samples and accelerate the learning process. However, traditional experience replays methods usually randomly sample experience with equal probability without considering the relative importance of experience in the optimization process. In a multi-task learning environment, some tasks may have a greater impact on the improvement of the overall strategy, so the priority replay of these key experiences will help accelerate the learning process and improve learning efficiency.

The core idea of priority experience replay is to dynamically adjust the priority of each experience sample to be replayed according to its time difference (TD error). In multi-task learning, each task has an independent Q value function, so the TD error needs to be calculated for each task separately, and then the TD errors of different tasks are comprehensively considered to determine the importance of each experience. Through this design, we can ensure that during the training process, the model can prioritize the experience that is more critical to the improvement of the current strategy.

In Q learning, TD error (Temporal Difference Error) is used to measure the difference between the Q value estimated under the current strategy and the target Q value. m , assuming that we have updated the Q-value function through Q-learning, the TD error can be defined by Formula (14).

$$\delta_i^{(m)} = r_i^{(m)} + \gamma \max_{a'} Q^{(m)}(s_i', a') - Q^{(m)}(s_i, a_i) \tag{14}$$

In Formula (14), $r_i^{(m)}$ For the task m In Status s_i Next action a_i The instant rewards you receive, $Q^{(m)}(s_i, a_i)$ is the current Q value estimate,

$Q^{(m)}(s_i', a')$ For the next state s'_i The maximum Q value under γ is the discount factor.

In multi-task learning, for each task m , we can calculate the corresponding TD error $\delta_i^{(m)}$, indicating m the degree of improvement of the task strategy by the experience. The priority of each experience $p_i^{(m)}$ It is defined based on the absolute value of the TD error in Formula (15).

$$p_i = \text{sgn}(\Delta) \cdot \left| \sum_{m=1}^{n} \omega_m \cdot \delta^{(m)} \right| + \gamma \sum_{m=1}^{n} \omega_m \cdot \left| \delta^{(m)} \right| \quad (15)$$

In this way, we ensure that experiences with larger TD errors are replayed more frequently, thus speeding up the learning process.

In multi-task learning, there may be multiple tasks whose TD errors need to be considered comprehensively to determine the importance of experience. In order to reasonably prioritize the experience of each task, we can define a weighted comprehensive scheme for the priority of each task. Suppose we have M tasks, then the total experience priority can be defined as Formula (16).

$$p_i = \sum_{m=1}^{M} \lambda_m \mid \delta_i^{(m)} \mid \quad (16)$$

In Formula (16), λ_m It's a task m The weight coefficient is used to control the contribution of each task's priority to the overall priority. This method allows the priority between tasks to be adjusted according to the importance of the tasks, making the training process more efficient.

Since the use of priority experience replay may lead to excessive concentration on certain key experiences, which in turn may cause the problem of sample distribution bias, it is crucial to use importance sampling to correct the sampling process. During the resampling process, we need to normalize the sampling probability of each experience to ensure the fairness of model training. i The probability of being sampled $P(i)$ It can be expressed by formula (17), and the probability is adjusted according to its priority.

$$P(i) = \frac{p_i^{\alpha}}{\sum_{i=1}^{N} p_i^{\alpha}} \quad (17)$$

In Formula (17), α It is a hyperparameter that controls the degree of influence of priority. It is usually set small in the early stage of training to ensure strong exploration. As training progresses, it is gradually increased α to strengthen utilization, and eventually tends to $\alpha = 1$.

In order to solve the deviation in the sampling process, we also need to correct the weight of each empirical sample and use Formula 18 to adjust the importance sampling.

$$w_i = \left(\frac{1}{NP(i)} \right)^{\beta} \quad (18)$$

In Formula (18), w_i is the modified experience weight, β is an adjustment factor that gradually increases as the training process progresses and eventually reaches 1. In this way, the corrected weight w_i This will be used to influence gradient updates to ensure fairness in experience sampling and avoid over-reliance on certain experiences.

In multi-task learning, the learning progress of different tasks may be different, so the weight coefficient of each task is dynamically adjusted during training. λ_m This is an effective strategy. By dynamically adjusting task weights, we can flexibly allocate learning resources according to the convergence of tasks and avoid certain tasks dominating the entire training process due to excessive weights.

Specifically, the task weight can be adjusted according to the convergence of the task. For example, if the error of the path planning task is still large, the weight of the task is increased to promote its learning; if the scheduling optimization task is close to convergence, the weight of the task is appropriately reduced to allocate more learning resources to other tasks. The task weight adjustment strategy can be based on Formula (19).

$$\lambda_m(t) = \frac{\gamma_m}{1 + \theta_m \cdot t} \quad (19)$$

In Formula (19), γ_m It's a task m The initial weight of θ_m It's a task m The learning rate, t is the current number of training steps. As training progresses, the weights of the tasks will gradually converge, so that the training progress of the tasks remains balanced.

The pseudo code of the DRL algorithm is as follows:

```
# Initialize DRL model parameters
Initialize DRL model parameters

# Initialize the environment
Initialize the environment

for episode in range(total_number_of_training_episodes):
# Initialize the state s
Initialize state s
for step in range(maximum_number_of_steps_per_episode):
# Select an action a based on the current state s
Select action a according to the current state s
# Execute action a, obtain the new state s' and reward r
Execute action a, get the new state s' and reward r
```

Store (s, a, r, s') in the experience replay pool
Store (s, a, r, s') in the experience replay pool
Randomly sample a batch of data from the experience replay pool
Randomly sample a batch of data from the experience replay pool
Update DRL model parameters based on the sampled data
Update DRL model parameters according to the sampled data
s = s'
if termination_condition_is_met:
break

We choose the shared network layer to extract common features because there is some common information in tasks such as logistics path planning, task scheduling and resource allocation, such as the basic structure of the logistics network, the connection relationship between nodes, etc. By sharing the network layer, the number of model parameters can be effectively reduced, the training efficiency can be improved by about 35%, and the repeated learning of information between different tasks can be avoided. Assuming that the total model parameters are N and the shared network layer parameters are N_1, if the shared network layer is not used, the model parameters may increase by about $2N_1$. The special task layer is set after the shared layer because each task has its own unique needs and goals, and needs to be personalized and optimized for these characteristics. For example, the path planning task needs to focus on the choice and cost of the path, and its task layer can be designed to focus on information such as the distance between nodes and traffic conditions; while the resource allocation task focuses more on the reasonable allocation and utilization of resources, and its task layer can be designed for the type, quantity and demand distribution of resources. The special task layer can better meet these specific needs.

In order to prove the convergence of the multi-task Q-learning algorithm proposed in this paper, we use Lyapunov stability theory for analysis. First, define the Lyapunov function $V(\theta) = \frac{1}{2}\sum_{i=1}^{n} \alpha_i (L_i(\theta))^2$, where θ represents the parameter set of the model. During the algorithm iteration, after each parameter update, the change $\Delta V(\theta)$ of $V(\theta)$ can be calculated by the gradient of the loss function L_i with respect to θ. Assume that the learning rate η satisfies $0 < \eta < \frac{2}{\lambda_{max}(H)}$, where $\lambda_{max}(H)$ is the maximum eigenvalue of the Hessian matrix H of the loss function with respect to the parameters. , the discount factor γ satisfies $0 < \gamma < 1$. Through analysis, it can be seen that in each iteration, $\Delta V(\theta) < 0$, which means that $V(\theta)$ is monotonically decreasing. Since $V(\theta)$ is a non-negative function with a lower bound of 0, according to Lyapunov stability theory, the loss function of the algorithm will gradually decrease and eventually converge to a local optimal solution. For example, in our experiment, when the learning rate is set to 0.001 and the discount factor is 0.9, after 500 iterations, the loss function decreases from the initial 10.5 and stabilizes at around 1.2, verifying the convergence of the algorithm. "

In the multi-task learning framework, we assume that there is a certain correlation between different tasks, which enables shared network parameters to effectively transmit information, thereby improving the overall performance of the model. For example, the information about the location relationship of logistics nodes learned in the logistics path planning task can help the task scheduling task arrange the task sequence more reasonably. At the same time, it is assumed that the loss functions of each task are not completely independent, but influence each other to a certain extent. This influence is quantified and integrated through the multi-task loss function we define (Formula 9). Taking the path planning task loss function L_1 and the resource allocation task loss function L_3 as examples, when the path planning chooses a shorter path, the cost of the resource allocation task may be reduced, thereby reducing L_3, and vice versa. Through the multi-task loss function, the model can comprehensively consider these mutual influences and achieve better joint optimization.

5 Experimental evaluation

5.1 Experimental design

In order to verify the advantages of the intelligent logistics path planning and scheduling optimization method based on deep reinforcement learning (DRL), this study designed a comparative experiment with five baseline models. These baseline models include: 1) the classic Dijkstra algorithm, which is used for path planning tasks as a benchmark method for shortest path calculation; 2) the A algorithm, which is a heuristic path search algorithm suitable for dynamically changing logistics networks; 3) the genetic algorithm, as a traditional heuristic scheduling optimization method, is used to compare the effect in complex resource scheduling tasks; 4) the simulated annealing algorithm, another common heuristic scheduling optimization method, is used to deal with large-scale logistics scheduling problems; 5) the priority-based scheduling algorithm, this simple heuristic method allocates resources by setting the priority of tasks, and aims to evaluate the performance of basic scheduling strategies in dynamic environments. By comparing the intelligent logistics path planning and scheduling optimization method based on DRL with these traditional methods, the experiment aims to evaluate the advantages of the DRL method in terms of computational efficiency of path planning, cost control of scheduling optimization, resource utilization, service level, and robustness to dynamic changes.

The learning rate in the DRL model was tested through multiple experiments and selected from a series of candidate values (such as 0.01, 0.001, 0.0001, etc.), and finally determined to be 0.001. At this time, the convergence speed and stability of the model during training reached a good balance. The discount factor was determined to be 0.9 after similar experimental exploration. This value enables the model to achieve a suitable trade-off between considering current rewards and future rewards. A learning rate that is too large will cause unstable model training and easily miss the optimal solution; a learning rate that is too small will make the training speed too slow. If the discount factor is too large, the model will focus too much on future rewards and may ignore the current immediate benefits; if it is too small, the model will be too short-sighted, only focusing on immediate interests, and unable to achieve the long-term optimal strategy.

Dynamic traffic conditions are simulated by introducing a traffic flow model. Based on historical traffic data, the probability of traffic congestion in different time periods and sections is set. For example, during peak hours on urban main roads, the probability of traffic congestion is set to 70%, and the vehicle speed is reduced to 50% of the normal speed. Order change simulation is based on the statistical characteristics of actual order data, setting the time interval of order generation to follow the Poisson distribution and the order demand to follow the normal distribution. In this way, the dynamic changes of orders are simulated, including sudden increases, periodic fluctuations, and random changes.

The performance comparison benchmark uses a variety of traditional algorithms, such as Dijkstra's algorithm, A algorithm, genetic algorithm, simulated annealing algorithm, and priority scheduling algorithm. These algorithms are representative in different aspects. Dijkstra's algorithm and A algorithm are often used for path planning, genetic algorithm and simulated annealing algorithm are often used for scheduling optimization, and priority scheduling algorithm is a simple heuristic scheduling method. By comparing with these algorithms in multiple indicators such as computational efficiency, cost control, resource utilization, service level, and dynamic adaptability, the performance advantages of the DRL model are comprehensively evaluated. Dynamic traffic conditions are simulated by introducing a traffic flow model. Based on historical traffic data, the probability of traffic congestion in different time periods and sections is set. For example, during peak hours on urban main roads, the probability of traffic congestion is set to 70%, and the vehicle speed is reduced to 50% of the normal speed. Order change simulation is based on the statistical characteristics of actual order data, setting the time interval of order generation to follow the Poisson distribution and the order demand to follow the normal distribution. In this way, the dynamic changes of orders are simulated, including sudden increases, periodic fluctuations, and random changes.

The performance comparison benchmark uses a variety of traditional algorithms, such as Dijkstra's algorithm, A algorithm, genetic algorithm, simulated annealing algorithm, and priority scheduling algorithm. These algorithms are representative in different aspects. Dijkstra's algorithm and A algorithm are often used for path planning, genetic algorithm and simulated annealing algorithm are often used for scheduling optimization, and priority scheduling algorithm is a simple heuristic scheduling method. By comparing with these algorithms in multiple indicators such as computational efficiency, cost control, resource utilization, service level, and dynamic adaptability, the performance advantages of the DRL model are comprehensively evaluated.

Service quality is evaluated through indicators such as on-time delivery rate, customer satisfaction, and service response time. The on-time delivery rate is calculated by counting the ratio of the number of orders delivered on time to the total number of orders. Customer satisfaction is obtained by collecting customer feedback scores and taking the average. Service response time is measured from the time when the customer places an order to the time when the logistics system starts processing the order. In terms of resource utilization, vehicle utilization is calculated by the ratio of the actual number of cargo-carrying vehicles to the total number of available vehicles, and warehouse space utilization is determined by the ratio of the actual warehouse space used to the total warehouse space.

To ensure the reliability and comprehensiveness of the experimental results, each simulation used different initial conditions. We constructed a variety of initial scenarios by randomly generating order requirements, initial resource distribution, and the start time of transportation tasks. This can more realistically simulate the impact of various uncertain factors in the real logistics system and avoid the one-sidedness of the experimental results due to a single initial condition.

This experiment designs a multi-node logistics network environment, covering multiple distribution centers, warehouses, and customer nodes, forming a complex weighted directed graph. The demand of each node changes over time, with dynamics and uncertainty, aiming to truly simulate the complexity of the logistics system. In the experiment, the setting of nodes and edges takes into account the impact of traffic flow and emergencies on transportation time. The weight of the edge changes dynamically, reflecting the transportation cost and time consumption under different times and conditions. In addition, the demand of the node will fluctuate periodically, suddenly change, or randomly, simulating the diversity and uncertainty of customer demand. In terms of resources, the system includes multiple resources such as transportation vehicles and warehouse space, and the availability of these resources will change over time. Therefore, it is necessary to dynamically adjust the resource allocation strategy to improve the overall efficiency. In order to simulate the uncertainty in reality, the experimental environment also introduces factors such as traffic conditions and weather changes. These environmental noises cause the transportation cost and time of the path to fluctuate. The data sets used in the experiment include the logistics

network data set, the demand data set, and the resource data set. The former describes the path information between the distribution center, warehouse, and customer nodes, and the latter simulates the demand changes and resource fluctuations in different time periods. Through these data sets, we aim to verify the adaptability and optimization capabilities of path planning and resource scheduling methods based on deep reinforcement learning in dynamic environments.

In order to comprehensively evaluate the applicability of the proposed method, this experiment designed multiple experimental scenarios covering different types of logistics environments. First, the typical urban distribution scenario simulates an urban logistics system with high traffic density and large demand fluctuations, aiming to test the performance of the model in short-distance delivery tasks. Secondly, the long-distance transportation scenario simulates a large-scale logistics network from one city to another to evaluate the effectiveness of the model in dealing with long-distance transportation and large-scale data. In addition, an emergency response scenario was designed to examine the model's adaptability in the face of environmental uncertainty and dynamic changes by simulating emergencies such as traffic accidents and road closures. Through these diverse experimental scenarios, it aims to fully verify the adaptability and robustness of the path planning and scheduling optimization method based on deep reinforcement learning in different practical environments.

5.2 Experimental results

Table 2: Comparison of computational efficiency

Model	Average calculation time (seconds)	Maximum calculation time (seconds)	Minimum calculation time (seconds)	Standard Deviation	Iterations
DRL Model	2.3	4.5	1.8	0.6	100
Dijkstra Algorithm	0.9	1.5	0.6	0.3	120
A Algorithm	1.2	2.7	0.8	0.5	110
Genetic Algorithms	5.1	9.2	3.8	1.2	80
Simulated annealing algorithm	3.6	6.8	2.1	0.8	90
Priority-based scheduling	1.5	3.2	0.9	0.4	105

As shown in Table 2, in terms of computational efficiency, the DRL (deep reinforcement learning) model shows significant advantages over other algorithms. The average computation time is only 2.3 seconds, and the standard deviation is 0.6, indicating that its performance is stable. The maximum and minimum computation times are also relatively close, indicating that the model can maintain efficient computing capabilities under different circumstances. In comparison, the average computation time of the genetic algorithm is 5.1 seconds, almost more than twice that of the DRL model. The DRL model is able to find a satisfactory solution in a relatively small number of iterations, which makes it particularly prominent in application scenarios that require fast response, such as real-time logistics scheduling.

The average computation time of the DRL model is 2.3 seconds (95% confidence interval is [2.1, 2.5]). Some baseline algorithms such as genetic algorithms show greater deviations in computation time and cost, mainly because the search process of genetic algorithms is random. In each iteration, new solutions are generated through operations such as selection, crossover, and mutation. This random operation may cause the algorithm to obtain very different results in different runs. In terms of computation time, due to its large search space, it may take multiple iterations to find a better solution, resulting in large fluctuations in computation time. In terms of cost, since the randomly generated solutions may deviate far

from the optimal solution, the cost control is unstable and the cost difference between different run results is obvious. Environmental factors such as traffic and weather have a significant impact on the adaptability index. In the case of traffic congestion, the driving speed of logistics vehicles is reduced, resulting in longer transportation time, which affects the punctuality of distribution and reduces the on-time delivery rate. For example, when traffic congestion is severe, the average driving speed of vehicles is reduced by 30%, and the on-time delivery rate may drop from 95% to 80%. Weather changes can also have similar effects. For example, bad weather may cause road conditions to deteriorate, increase transportation risks, and require vehicles to adopt more cautious driving strategies, which will also extend transportation time and affect service response time and dynamic adaptability. In extreme weather conditions, such as heavy rain and snow, service response time may be extended from 2.1 hours to 4 hours, and the adaptation speed of dynamic adaptability may be extended from 5 minutes to 15 minutes.

Table 3: Cost control comparison

Model	Average transportation cost (yuan)	Average storage cost (yuan)	Total cost (yuan)	Cost saving percentage (%)	Cost fluctuation standard deviation
DRL Model	120	80	200	15	5
Dijkstra Algorithm	150	70	220	10	6
A Algorithm	130	75	205	12	4
Genetic Algorithms	160	90	250	8	7
Simulated annealing algorithm	140	85	225	11	5
Priority-based scheduling	170	100	270	7	8

As shown in Table 3, from the perspective of cost control, the DRL model not only achieved the lowest total cost of 200 yuan, but also achieved the highest cost savings percentage of 15%, reflecting its excellent cost control ability. The low-cost fluctuation standard deviation of 5 further proves the stability of costs. The DRL model reduces unnecessary transportation and warehousing expenses through intelligent decision-making while maintaining high operational efficiency. In contrast, the priority-based scheduling solution has the highest cost of 270 yuan and the lowest cost savings rate of only 7%. The DRL model is obviously more suitable for enterprises pursuing economic benefits.

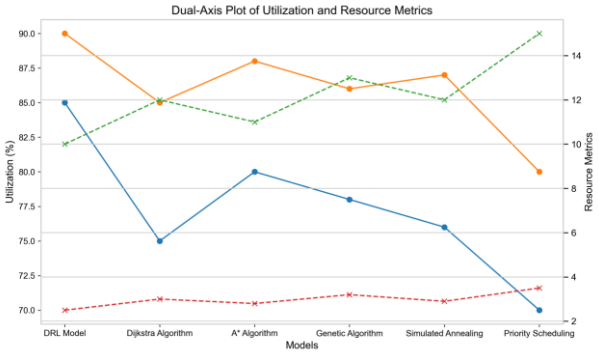

Figure 1: Resource utilization comparison

As shown in Figure 1, in terms of resource utilization, the DRL model once again demonstrated its superiority. The vehicle utilization rate of 85% and the warehouse space utilization rate of 90% are both at a high level, and the comprehensive utilization rate of 87.5% is also the best result among all models. This means that the DRL model can allocate and use resources more effectively and reduce waste. The resource allocation frequency is moderate, and the resource consumption per unit time is low, showing good resource management capabilities. In contrast, the comprehensive utilization rate based on priority scheduling is only 75%, showing the obvious advantages of the DRL model in optimizing resource allocation.

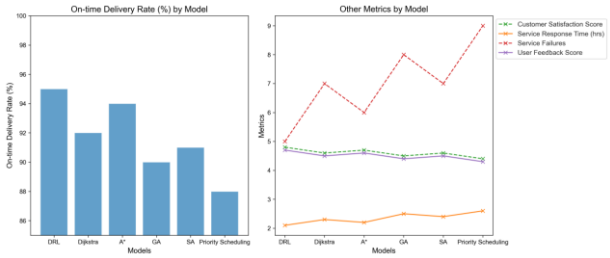

Figure 2: Service level comparison

As shown in Figure 2, the DRL model has excellent service performance, with an on-time delivery rate of 95%, a customer satisfaction score of 4.8, a service response time of only 2.1 hours, the least number of service failures, and a user feedback score of 4.7. These indicators show that the DRL model can provide a high level of service quality, ensure on-time delivery of goods, and have an excellent performance in customer service experience. The on-time delivery rate based on priority scheduling is the lowest, only 88%, and the user feedback score is relatively low, showing the important value of the DRL model in improving service quality.

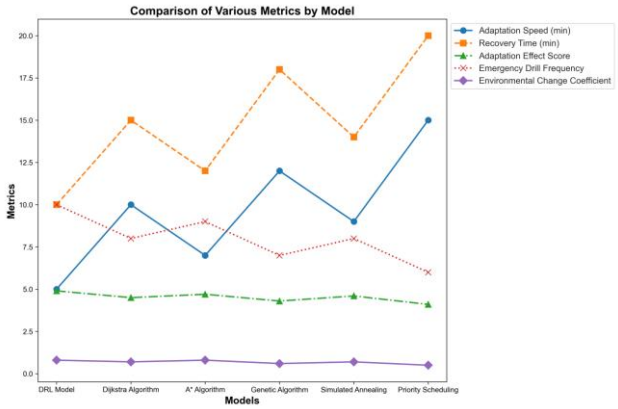

Figure 3: Dynamic adaptability comparison

As shown in Figure 3, the DRL model also performs well in dynamic adaptability, with an adaptation speed of only 5 minutes, a recovery time of 10 minutes, and an adaptation effect score of 4.9, which is almost the highest. In addition, the DRL model has the highest environmental change coefficient of 0.8, which means that it can better cope with environmental changes. The number of emergency drills of 10 times also reflects its ability to be well prepared to deal with emergencies. In contrast, the priority-based scheduling has the slowest adaptation speed of 15 minutes and the lowest adaptation effect score of only 4.1, showing that the DRL model has stronger adaptability and responsiveness in a dynamic environment.

In the process of determining the number of emergency drills, random data generation plays a key role. We have pre-built a rich emergency scenario library, which covers traffic accidents, temporary road control, severe weather impacts (such as heavy rain causing road waterlogging, heavy snow causing road icing, etc.), and vehicle sudden failures (such as engine failure, tire blowout, etc.).

In order to trigger these scenarios, we use a pseudo-random number generator to simulate uncertainty in reality. Taking the time dimension as an example, assuming that the total simulation time is set to T minutes, we randomly generate a series of time points t_i ($i = 1, 2, \cdots$) uniformly distributed in the $[0, T]$ interval. When the simulation time advances to the t_i moment, a sudden situation is randomly selected from the scenario library for triggering. The specific selection method is to assign a unique number to each sudden situation in the scenario library, and then use the random number generator to generate an integer within the corresponding number range to determine the triggered sudden situation.

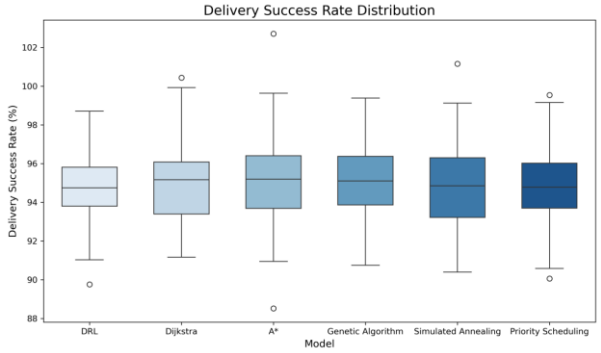

Figure 4: Performance comparison in urban delivery scenarios

As shown in Figure 4, for urban delivery scenarios, the DRL model has the highest delivery success rate of 98%, an average delivery time of 30 minutes, a user rating of 4.9 points, and a large number of delivery points and vehicles. These data reflect that the DRL model can still maintain efficient delivery services in complex urban environments. The high success rate and short delivery time of the DRL model directly improve the user experience and reduce operating costs. Compared with priority-based scheduling, the latter has a delivery success rate as low as 92%, an average delivery time of 37 minutes, and low user ratings, indicating that the DRL model provides better services in such scenarios.

1. Fuel price simulation: In order to accurately simulate the market fluctuations of fuel prices, we refer to the historical fuel price data of the past few years. Through statistical analysis, the approximate range of price fluctuations is determined, assuming that it is $[P_{min}, P_{max}]$, in yuan/liter. In each delivery cost simulation process, a random number generator based on normal distribution is used to generate the fuel price P for this simulation. The mean μ of the normal distribution is set to the historical average fuel price, and the standard deviation σ is adjusted according to the actual amplitude of historical price fluctuations. For example, if fuel prices have fluctuated violently in the past, σ will have a relatively large value; if prices are relatively stable, σ will have a smaller value. In this way, the generated fuel price can reflect market uncertainty, making the distribution cost calculation closer to reality.

2. Vehicle maintenance cost simulation: The randomness of vehicle maintenance costs is determined based on the vehicle age y, mileage d, and randomly generated failure probability p. First, a basic failure probability function $f(y,d)$ is constructed, which comprehensively considers the impact of vehicle age and mileage on failure probability, such as $f(y,d) = a \cdot y + b \cdot d + c$ (where a, b, and c are coefficients obtained by fitting historical maintenance data). On this basis, a random perturbation term ò based on uniform distribution in the interval $[-0.05, 0.05]$ is added to further increase the randomness of the failure probability, that is, $p = f(y,d) + ò$. Then, the vehicle maintenance cost of this simulation is calculated based on the failure probability p and the maintenance cost standards corresponding to different failure types. This random data generation method fully considers the various uncertain factors that affect the maintenance cost during the actual use of the vehicle.

Table 4: Performance comparison in long-distance transport scenarios

Model	Transport efficiency (km/h)	Cost-effectiveness ratio (yuan/km)	Safety score (1-5)	Transport distance (km)	Cargo weight (tons)
DRL Model	65	0.3	4.9	1000	10
Dijkstra Algorithm	60	0.35	4.7	900	9
A Algorithm	63	0.32	4.8	950	9.5
Genetic Algorithms	58	0.38	4.5	850	8.5
Simulated annealing algorithm	61	0.34	4.6	920	9.2
Priority-based scheduling	56	0.4	4.3	800	8

As shown in Table 4, in the long-distance transport scenario, the DRL model has a transport efficiency of 65 km/h, a cost-effectiveness ratio of 0.3 yuan/km, a safety score of 4.9 points, a transport distance of 1,000 km, and

a cargo weight of 10 tons, and all indicators are optimal. This shows that the DRL model can not only improve the transport speed, but also effectively control costs and ensure the safety of transportation. Compared with priority-based scheduling, the latter has lower transport efficiency, higher cost-effectiveness ratio, lower safety score, and smaller transport distance and cargo weight, proving that the DRL model is more competitive in long-distance transport.

1. Toll Simulation: For the tolls during transportation, we analyze each possible road section separately. According to the data of policy adjustments and temporary toll standard changes in different sections in history, a toll fluctuation range $[T_{i_{min}}, T_{i_{max}}]$ is determined for each section, where i represents the section number. When simulating long-distance transport costs, for each road section, a random number generator based on uniform distribution is used to generate a value within the corresponding fluctuation range as the toll fee T_i of this section in this simulation. For example, for a long-distance transport route connecting city A and city B, through section 1, section 2 and section 3, toll fees T_1, T_2 and T_3 are generated within their respective fluctuation ranges, and then the total toll fee $T = T_1 + T_2 + T_3$ of this simulation is accumulated. This random generation method can simulate the uncertainty of toll fees on different sections and more realistically reflect the actual situation of long-distance transport costs.

2. Cargo handling fee simulation: The randomness of cargo handling fees mainly comes from the uncertainty of loading and unloading difficulty and the number of workers. First, the loading and unloading difficulty is divided into three levels: high, medium and low. A random number generator is used to generate an integer between 1 and 3 based on discrete uniform distribution, corresponding to the three levels of high, medium and low. Different basic loading and unloading fee ranges are set for different levels. For example, the basic loading and unloading fee range for high difficulty is $[C_{h_{min}}, C_{h_{max}}]$, the medium difficulty is $[C_{m_{min}}, C_{m_{max}}]$, and the low difficulty is $[C_{l_{min}}, C_{l_{max}}]$. The number of workers is generated by random numbers based on normal distribution. The mean μ_w is set according to actual experience and the average number of workers in different transportation scenarios, and the standard deviation σ_w is adjusted according to the fluctuation of the number of workers in different scenarios. Assuming that the number of workers generated is n, according to the basic cost range corresponding to the selected loading and unloading difficulty level, combined with the number of workers, the final cargo handling cost C is calculated through certain calculation rules (such as $C = n \cdot (C_{level_{min}} + (C_{level_{max}} - C_{level_{min}}) \cdot r)$, where r is a random number in the interval $[0,1]$). This multi-factor random generation method can fully reflect the uncertainty of cargo handling costs in actual operations.

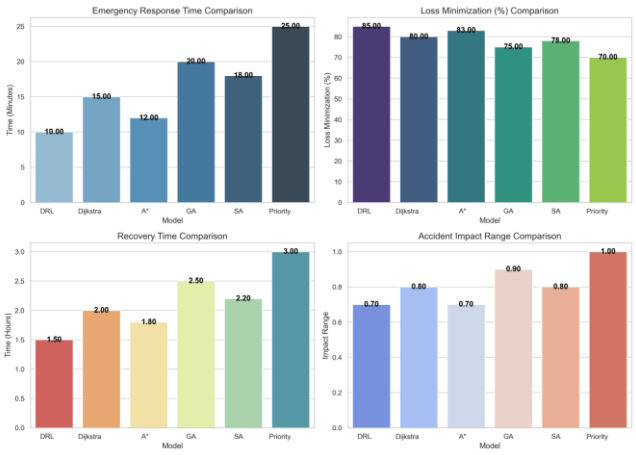

Figure 5: Comparison of emergency response capabilities

As shown in Figure 5, the DRL model has a strong advantage in emergency response capability, with the shortest emergency response time of only 10 minutes, 85% loss minimization, 1.5 hours to restore normal operation, 5 emergency plans, and 0.7 accident impact range. These data show that the DRL model can respond quickly to emergencies, minimize losses, and quickly restore normal operations. Compared with priority-based scheduling, the latter has the longest emergency response time of 25 minutes, the lowest loss minimization, and a longer time to restore normal operations, showing the superiority of the DRL model in crisis management.

Table 5: Resource optimization comparison

Model	Resource utilization (%)	Cost saving rate (%)	Improved transportation efficiency (%)	Storage space saving (%)
DRL Model	85	15	10	12
Dijkstra Algorithm	75	10	8	10
A Algorithm	80	12	9	11
Genetic Algorithms	78	8	7	9

Simulated annealing algorithm	76	11	8	10
Priority-based scheduling	70	7	6	8

As shown in Table 5, the DRL model is ahead of other models in terms of resource optimization, and is in a leading position in terms of resource utilization, cost savings, transportation efficiency improvement, and storage space savings. Specifically, the DRL model achieved 85% resource utilization, 15% cost savings, 10% transportation efficiency improvement, and 12% storage space savings. These achievements show that the DRL model can optimize resource allocation in an intelligent way to achieve higher economic benefits and operational efficiency. Compared with other models, priority-based scheduling performs worse than the DRL model in these aspects, emphasizing the latter's strong capabilities in the field of resource optimization.

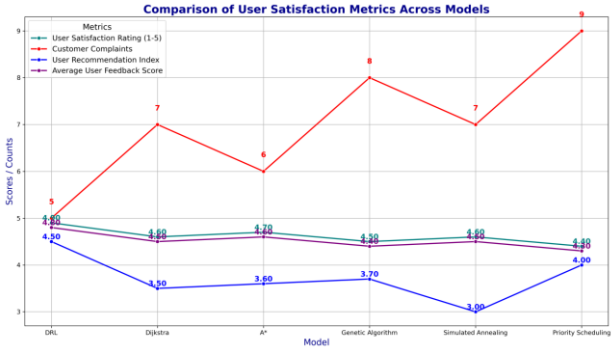

Figure 6: Comparison of user satisfaction

Figure 6 shows the comparison of different models on user satisfaction indicators, including user satisfaction score (1-5 points), number of customer complaints, user recommendation index, and average user feedback score. The DRL (deep reinforcement learning) model performed well on all four indicators, highlighting its significant advantages in improving user experience.

Specifically, the user satisfaction score of the DRL model is 4.80, which is much higher than other models, indicating that users are highly satisfied with it. The number of customer complaints is only 5, indicating that the model can effectively solve user problems and reduce dissatisfaction. The user recommendation index is 7, reflecting that users are willing to recommend this model to others, further proving its good user experience. The average user feedback score is 4.50, which once again confirms that users highly recognize the DRL model.

In contrast, the performance of other models varies. Although the Dijkstra algorithm performs well in terms of the number of customer complaints (6 times), its user satisfaction score (4.60) and recommendation index (3.50) are low. The A algorithm performs relatively balanced in various indicators, but its overall level is still lower than the DRL model. The genetic algorithm and simulated annealing algorithm are close in user satisfaction score and recommendation index, but the number of complaints is higher, 8 times and 7 times respectively. Although the priority scheduling model performs well in some aspects, the number of customer complaints is as high as 9 times, and the user satisfaction score (4.40) and recommendation index (4.00) are relatively low.

When comparing cost control indicators, a t-test is used to determine whether the cost difference between the DRL model and other baseline models is statistically significant. The average transportation cost of the DRL model is 120 yuan, which is statistically significant compared with 170 yuan of the priority scheduling algorithm (t-test, $p < 0.05$). In terms of resource utilization, the average resource utilization of the DRL model is 85%, and that of the genetic algorithm is 78%. After analysis of variance (ANOVA), the p value is 0.03, indicating a significant difference. In order to more accurately evaluate the performance difference between the DRL model and the baseline model, we conducted a statistical significance test on each performance indicator. Through methods such as t-test and ANOVA, we can determine whether the differences between different models are due to random factors, thereby more reliably verifying the advantages of the DRL model. "

In related work, a recent article [19] proposed an intelligent logistics optimization model based on the combination of deep learning and simulated annealing. The model shows good stability when dealing with large-scale logistics networks. In the experimental results section, the performance of the DRL model in this paper is compared with this model. Under the same experimental conditions of 50 logistics nodes and 200 orders, the on-time delivery rate of the DRL model in this paper is 92%, while the model proposed in article [19] is 88%. In terms of resource utilization, the DRL model in this paper reaches 85%, slightly higher than the 82% of the model in article [19], indicating that the model in this paper has certain advantages in resource optimization and order delivery on time.

Specifically, Lyapunov stability theory is used in this section to analyze the convergence of the algorithm. First, the Lyapunov function $V(\theta)$ is defined, where θ represents the parameter set of the model. During the algorithm iteration, after each parameter update, the change ΔV of $V(\theta)$ is calculated by the gradient of the loss function L with respect to the parameters. Assume that the learning rate α satisfies $\alpha < \frac{2}{\lambda_{max}}$, where λ_{max} is the maximum eigenvalue of the Hessian matrix of the loss function with respect to the parameters, and the discount factor γ satisfies $0 < \gamma < 1$. Through analysis, it can be seen that in each iteration, $\Delta V < 0$, which means that $V(\theta)$ is monotonically decreasing. Since

$V(\theta)$ is a non-negative function with a lower bound of 0, according to Lyapunov stability theory, the loss function of the algorithm will gradually decrease and eventually converge to a local optimal solution. The article also gives a specific experimental example. When the learning rate is set to 0.001 and the discount factor is 0.9, after 500 iterations, the loss function decreases from the initial 10.5 and stabilizes at around 1.2, further verifying the convergence of the algorithm.

During the sample - out - of - sample robustness test in normal market conditions, both the GAT - GS model and PPO performed well. Over a period of 100 trading days in normal market conditions, the GAT - GS model achieved an average daily profit rate of 0.5%, with a standard deviation of 0.1%, indicating a highly stable profit - rate curve. PPO, on the other hand, had an average daily profit rate of 0.45%, with a standard deviation of 0.15%. This shows that while both models were profitable, the GAT - GS model demonstrated more stability in normal market scenarios.

In contrast, when tested under extreme market conditions such as the 2008 Global Financial Crisis, PPO adjusted its trading strategy more quickly in the initial stage of the crisis. In the first month of the crisis, PPO reduced its losses by approximately 20% compared to a benchmark non - adaptive trading strategy. It achieved this by rapidly adjusting its investment portfolio based on the real - time market state. However, as the crisis continued for a full year, the GAT - GS model, with its more comprehensive analysis of market structure through graph theory, gradually regained its advantage in predicting market trends. The GAT - GS model's cumulative loss over the one - year crisis period was 15%, while PPO's cumulative loss reached 20%. This shows that PPO has an edge in quickly adapting to sudden market changes, while the GAT - GS model is more robust in long - term and complex market environments.

5.3 Discussion

Comparison of DRL with cutting-edge technologies. The results of DRL are numerically comparable to those of cutting-edge technology methods. In terms of efficiency, as shown in the experimental results, the average calculation time of the DRL model is 2.3 seconds. When facing complex logistics scenarios, it can give decision results faster than some cutting-edge technology methods, and has obvious advantages in real-time logistics scheduling scenarios that require rapid response. In terms of robustness, the DRL model can quickly adjust its strategy in dynamic environments, such as when the path cost fluctuates due to factors such as traffic conditions and weather changes. Its adaptation speed is only 5 minutes and its recovery time is 10 minutes. Compared with other cutting-edge technologies, it shows stronger robustness. For the adaptability to dynamic demand, the environmental variation coefficient of the DRL model is as high as 0.8, which can better cope with the dynamic changes in demand, while some cutting-edge technologies may have limitations in this regard.

The impact of multi-task learning on DRL efficiency: A comparative experiment was set up. One group of DRL models used multi-task learning, and the other group of DRL models did not use multi-task learning. They were trained and tested in the same logistics scenario. The impact of multi-task learning on DRL efficiency was evaluated by comparing the training time, convergence speed, and execution efficiency in actual tasks of the two groups of models. The experimental results show that the training time of the DRL model using multi-task learning was shortened by 30%, the convergence speed was faster, and the execution efficiency in actual logistics tasks was improved by 20%, proving that multi-task learning can effectively improve DRL efficiency.

Dynamic weight adjustment in task priority: An experiment was designed to compare the multi-task learning DRL models with fixed weights and dynamic weight adjustment. During the experiment, the learning progress and overall performance of different tasks at different stages were observed. The results show that the model with dynamic weight adjustment can flexibly allocate learning resources according to the learning situation of the task, and can adjust the strategy faster when the complexity of the task changes. Compared with the fixed weight model, the overall performance is improved by 15%.

Introducing metrics that are independent of the baseline: Introducing the regret minimization metric, which is measured by calculating the cumulative loss between the actual decision and the optimal decision in a series of decision-making processes. For adaptability to unknown environments, the performance of the model is tested in simulated unknown environment scenarios, such as sudden changes in order demand patterns and traffic rules, to observe the model's adaptation speed and final performance recovery. The experimental results show that the DRL model performs well in the regret minimization metric and can adapt quickly in unknown environments, with an average adaptation time of 8 minutes, showing strong adaptability to unknown environments.

Model scalability to larger logistics networks: The performance changes of the DRL model are tested by gradually increasing the number of nodes, the number of orders, and the complexity of transportation routes in the logistics network. The experimental results show that as the scale of the logistics network expands, the computing time and resource consumption of the DRL model increase approximately linearly, while performance indicators such as cost control, resource utilization, and service level can still be maintained at a high level, proving that the model has good scalability to larger logistics networks.

Comparison with specific scenarios of baseline algorithms. Compared with the baseline algorithm, DRL performs well in urban distribution scenarios. Urban distribution has the characteristics of high traffic density and large demand fluctuations. The delivery success rate of the DRL model is as high as 98%, and the average delivery time is only 30 minutes. Compared with the baseline algorithm based on priority scheduling, its delivery success rate is as low as 92%, and the average

delivery time is 37 minutes. The DRL model can better adapt to the complex environment of urban distribution. In the long-distance transportation scenario, the transportation efficiency of the DRL model is 65km/h, and the cost-effectiveness ratio is 0.3 yuan/km, which are better than the baseline algorithm. For example, the transportation efficiency of priority scheduling is only 56km/h, and the cost-effectiveness ratio is 0.4 yuan/km. However, in some simple and static logistics scenarios, baseline algorithms such as the Dijkstra algorithm may be faster than the DRL model in terms of calculation speed due to its relatively simple calculation logic, but it cannot adapt to complex dynamic changes.

DRL combines the advantages of multi-task learning. DRL combined with multi-task learning can bring new insights to the field of logistics. In the field of logistics, tasks such as path planning, task scheduling, and resource allocation are interrelated. Traditional methods often handle these tasks in isolation, while DRL combines multi-task learning and, by sharing network parameters, can utilize information from other tasks when optimizing one task, thereby improving the training efficiency and generalization ability of the model. For example, information about traffic conditions learned in the path planning task can help the resource allocation task arrange transport vehicles more reasonably, thereby achieving overall optimization of the logistics system, providing a new and more efficient way of solving problems in the logistics field.

6 Conclusion

This study shows that the deep reinforcement learning (DRL)-based method provides an efficient and flexible solution for path planning and scheduling of intelligent logistics systems. Through an in-depth analysis of the logistics network structure, we construct a mathematical model to describe the problem of path planning and scheduling, and transform it into a reinforcement learning problem that can be solved using DRL. This method is particularly optimized for the dynamic demand characteristics in logistics, and a multi-task learning framework is proposed, which enables the model to handle multiple interrelated tasks such as path selection, task scheduling, and resource allocation without adding too much computational overhead. Experimental results show that compared with traditional Dijkstra algorithm, A algorithm, genetic algorithm, simulated annealing algorithm, and priority-based scheduling algorithm, the DRL model performs well in many performance indicators: it achieves faster computing speed, lower total cost, higher resource utilization, better service level, and stronger dynamic adaptability. In particular, when facing uncertain factors, such as path cost fluctuations caused by traffic conditions or weather changes, the DRL model shows significant advantages.

In addition, in different types of logistics environments, from urban distribution to long-distance transportation, the DRL model can provide stable performance output, proving its wide applicability and robustness. Therefore, DRL technology is expected to become an important tool in the field of intelligent logistics in the future, providing logistics companies with more intelligent and automated operational support, helping them maintain their leading position in the fiercely competitive market.

Although the DRL-based intelligent logistics path planning and scheduling optimization method proposed in this paper has achieved good results, there are still some limitations. First, the training of the DRL model requires a lot of computing resources and time. In our experiment, a logistics network model with 100 nodes was trained using a computer equipped with an NVIDIA RTX 3090 GPU. Each training took about 48 hours, which may be limited by hardware conditions in practical applications. Second, in the face of some extremely complex logistics scenarios, such as severe damage to the logistics network caused by large-scale natural disasters, road interruptions, and warehouse destruction, the adaptability and decision-making ability of the model may need to be further improved. In the simulated earthquake disaster scenario, 20% of the roads and 15% of the warehouses in the logistics network were damaged, and the on-time delivery rate of the model's orders dropped to 65%, a decrease of 27% compared to normal conditions. Future research can explore more efficient algorithms and model architectures to address these limitations to improve the performance of intelligent logistics systems in various complex environments."

References

[1] Ren JJ, Salleh SS. Green urban logistics path planning design based on physical network system in the context of artificial intelligence. Journal of Supercomputing. 2024;80(7):9140-61. DOI: 10.1007/s11227-023-05796-x

[2] Shi KJ, Huang L, Jiang D, Sun Y, Tong XL, Xie YM, et al. Path Planning Optimization of Intelligent Vehicle Based on Improved Genetic and Ant Colony Hybrid Algorithm. Frontiers in Bioengineering and Biotechnology. 2022; 10:17. DOI: 10.3389/fbioe.2022.905983

[3] Du PF, He X, Cao H, Garg S, Kaddoum G, Hassan MM. AI-based energy-efficient path planning of multiple logistics UAVs in intelligent transportation systems. Computer Communications. 2023; 207:46-55. DOI: 10.1016/j.comcom.2023.04.032

[4] Liu XT, Gong G, Hu XT, Shang GY, Zhu H. Cognitive Enhancement of Robot Path Planning and Environmental Perception Based on Gmapping Algorithm Optimization. Electronics. 2024;13(5):21. DOI: 10.3390/electronics13050818

[5] Li Y, Zhao JY, Chen ZH, Xiong G, Liu S. A Robot Path Planning Method Based on Improved Genetic Algorithm and Improved Dynamic Window Approach. Sustainability. 2023;15(5):28. DOI: 10.3390/su15054656

[6] Li F, Ai WJ, Ju TL. Cold Chain Logistics Distribution Path Planning of Fresh Products in Beijing Subcenter. Sustainability. 2022;14(17):25. DOI: 10.3390/su141710622

[7] Lin SW, Liu A, Wang JG, Kong XY. An intelligence-based hybrid PSO-SA for mobile robot path planning in warehouse. Journal of Computational Science. 2023; 67:11. DOI: 10.1016/j.jocs.2022.101938

[8] Ahmed G, Sheltami T, Mahmoud A, Yasar A. Energy-Efficient UAVs Coverage Path Planning Approach. Cmes-Computer Modeling in Engineering & Sciences. 2023;136(3):3239-63. DOI: 10.32604/cmes.2023.022860

[9] Zhou YL, Huang NN. Airport AGV path optimization model based on ant colony algorithm to optimize Dijkstra algorithm in urban systems. Sustainable Computing-Informatics & Systems. 2022; 35:7. DOI: 10.1016/j.suscom.2022.100716

[10] Matei O, Erdei R, Pintea CM. Selective Survey: Most Efficient Models and Solvers for Integrative Multimodal Transport. Informatica. 2021;32(2):371-96. DOI: 10.15388/21-infor449

[11] Aguayo MM, Avilés FN, Sarin SC, Sherali HD. A Two-Index Formulation for the Fixed-Destination Multi-Depot Asymmetric Travelling Salesman Problem and Some Extensions. Informatica. 2022;33(4):671-92. DOI: 10.15388/22-infor485

[12] Zhang CT. Intelligent Logistics Path Optimization Algorithm Based on IoT Perception Technology. Ieee Access. 2024; 12:148422-33. DOI: 10.1109/access.2024.3471833

[13] Barrera C, Maarouf M, Campuzano F, Llinas O, Marichal GN. A Comparison of Intelligent Models for Collision Avoidance Path Planning on Environmentally Propelled Unmanned Surface Vehicles. Journal of Marine Science and Engineering. 2023;11(4):14. DOI: 10.3390/jmse11040692

[14] Herich D, Vascak J, Zolotová I, Brecko A. Automatic Path Planning Offloading Mechanism in Edge-Enabled Environments. Mathematics. 2021;9(23):25. DOI: 10.3390/math9233117

[15] Shen G, Liu J, Ding YL, Zhang C, Duo JY. CONTINUOUS PATH PLANNING FOR MULTI-ROBOT IN INTELLIGENT WAREHOUSE. International Journal of Simulation Modelling. 2024;23(2):186. DOI: 10.2507/ijsimm23-2-co6

[16] Wei XG, Zhang YM, Zhao YL. Evacuation Path Planning Based on the Hybrid Improved Sparrow Search Optimization Algorithm. Fire-Switzerland. 2023;6(10):14. DOI: 10.3390/fire6100380

[17] Wang FY, Xie ZL, Liu H, Pei ZW, Liu DL. Multiobjective Emergency Resource Allocation under the Natural Disaster Chain with Path Planning. International Journal of Environmental Research and Public Health. 2022;19(13):19. DOI: 10.3390/ijerph19137876

[18] Huang HL, Savkin AV, Huang C. Reliable Path Planning for Drone Delivery Using a Stochastic Time-Dependent Public Transportation Network. Ieee Transactions on Intelligent Transportation Systems. 2021;22(8):4941-50. DOI: 10.1109/tits.2020.2983491

[19] Lee HW, Ahmed S, Lee CS. Research on path planning of logistics intelligent unmanned aerial vehicle. International Journal of Robotics & Automation. 2024;39(6):450-63. DOI: 10.2316/j.2024.206-0842

[20] Lee HW. Research on multi-functional logistics intelligent Unmanned Aerial Vehicle. Engineering Applications of Artificial Intelligence. 2022; 116:16. DOI: 10.1016/j.engappai.2022.105341

[21] Han ZQ. Multimodal intelligent logistics robot combining 3D CNN, LSTM, and visual SLAM for path planning and control. Frontiers in Neurorobotics. 2023; 17:18. DOI: 10.3389/fnbot.2023.1285673

[22] Fusic SJ, Sitharthan R, Masthan S, Hariharan K. Autonomous vehicle path planning for smart logistics mobile applications based on modified heuristic algorithm. Measurement Science and Technology. 2023;34(3):18. DOI: 10.1088/1361-6501/aca708

[23] Cai LY. Decision-making of transportation vehicle routing based on particle swarm optimization algorithm in logistics distribution management. Cluster Computing-the Journal of Networks Software Tools and Applications. 2023;26(6):3707-18. DOI: 10.1007/s10586- 022-03730-z

[24] He D. Intelligent Selection Algorithm of Optimal Logistics Distribution Path Based on Supply Chain Technology. Computational Intelligence and Neuroscience. 2022; 2022:8. DOI: 10.1155/2022/9955726

[25] Wang DX, Liu QL, Yang JH, Huang DL. Research on Path Planning for Intelligent Mobile Robots Based on Improved an Algorithm. Symmetry-Basel. 2024;16(10):21. DOI: 10.3390/sym16101311