Universiteit Leiden

Master Computer Science

Dynamic Courier Staffing for On-Demand Food Delivery with Distributional RL and Mixture-of-Experts

Name:	[Junze Yang]
Student ID:	[S4244532]
Date:	[15/01/2026]
Specialisation:	[Artificial intelligence]
1st supervisor:	[Yingjie Fan]
2nd supervisor:	[Thomas Bäck]

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

With the rapid growth of on-demand food delivery, platforms increasingly rely on crowdsourced couriers to meet promised delivery times. However, existing approaches often struggle to maintain robust performance when order volumes surge and courier availability fluctuates, especially at large scales.

To address this problem, we propose an adaptive uncertainty-aware scheduling framework that dynamically decides on-demand courier allocation and whether to extend active couriers' near-ending shifts.

Instead of using traditional Deep Q-Networks (DQN), we adopt Implicit Quantile Networks (IQN) to model the full return distribution, improving value estimation under environmental uncertainty.

To better handle heterogeneous operating conditions, we incorporate a mixture-of-experts (MoE) state encoder that adaptively routes input to experts across operational regimes.

We conduct large-scale courier scheduling experiments with up to 1,000 orders and show that our approach substantially improves overall performance and convergence efficiency over the baseline, indicating its potential for deployment in crowdsourced food-delivery operations.

Contents

1 Introduction

Courier staffing and scheduling in food delivery platforms (1; 2; 3; 4; 5) has gained increasing attention in the operations research community, given their material implications for platform profitability and service-level performance. These platforms require decision makers to optimize operations subject to service-level constraints while managing operational costs and courier-related considerations.

Although fixed courier scheduling, with shifts set in advance, offers operational predictability and cost controllability, it struggles to respond in real time to short-term demand fluctuations. Motivated by this limitation, this paper develops dynamic staffing strategies that improve operational efficiency while meeting service-reliability requirements under demand and supply uncertainty.

Traditionally, staffing and scheduling problems have been addressed using exact optimization methods or heuristic approaches. Exact methods (6) typically formulate the problem as mathematical programming models and employ robust optimization techniques to develop tractable optimization formulations for staffing and scheduling decisions. Heuristic approaches (7) such as adaptive large neighborhood search (ALNS) iteratively apply destroy and repair operators to explore the solution space and generate near-optimal solutions. However, these methods often struggle with complex multi-objective optimization problems involving conflicting objectives and large-scale solution spaces. To address these limitations, metaheuristic methods (8) have emerged, including hybrid algorithms that combine multiple optimization strategies such as NSGA-II and MOPSO with hypervolume-based adaptive search mechanisms to effectively optimize multiple conflicting objectives.

Despite their computational advantages, these traditional approaches are generally suboptimal in highly dynamic environments. In particular, many heuristics are developed under static or quasi-static problem formulations, in which order information is assumed to be known over a planning horizon, limiting their effectiveness in environments with stochastic arrivals and rapidly evolving system states. Moreover, such methods typically require re-optimization from scratch at each decision epoch, which introduces significant computational latency and often results in myopic decision policies that fail to anticipate future demand fluctuations.

Deep reinforcement learning (DRL) provides an effective framework for addressing complex sequential decision-making problems, in which agents parameterized by neural networks learn value functions directly from reward feedback and improve their decisions through interaction with the environment (9; 10; 11; 12; 13; 14).

However, traditional RL methods exhibit notable limitations when applied to dynamic courier staffing problem. On the one hand, expectation-based value learning fails to capture the heavy-tailed return distributions induced by stochastic demand surges, resulting in risk-insensitive and unstable decisions. On the other hand, standard RL architectures lack two essential capabilities for real-world delivery operations: the ability to specialize across different operational regimes (e.g., peak vs. off-peak hours), and the ability to handle delayed capacity responses inherent in courier onboarding processes.

Unlike traditional DRL methods that rely on expectation-based value learning and monolithic policy architectures, we formulate and propose an adaptive uncertainty-aware schedul-

ing framework, wherein distributional value estimates are directly learned to enable risk-sensitive decision-making and regime-specific policy specialization. Unlike existing expectation-based methods that feature simplified reward modeling, our framework not only achieves superior operational stability but also does so with minimal architectural complexity.

Our contributions are summarized as follows. (1) We propose an adaptive uncertainty-aware scheduling framework that explicitly models return distributions rather than point estimates, diverging from previous expectation-based methods that overlook operational risks. (2) We design three synergistic components to instantiate our framework. Particularly, we employ IQN (36) for distributional value estimation, integrate a MoE architecture (39; 38) for regime-adaptive policy routing, and introduce an Extend-Shift mechanism that enables immediate capacity adjustment by proactively extending near-ending couriers' shifts. (3) Extensive experimental results on large-scale courier scheduling scenarios with order volumes ranging from 200 to 1,000 show that our framework not only achieves superior average performance but also exhibits substantially improved operational stability, even outperforming expectation-based baselines during high-variance periods. Due to its effectiveness and practical interpretability, our distributional approach is expected to provide valuable insights for broader stochastic resource allocation domains, paving the way for deployable risk-aware workforce management systems.

2 Related Work

2.1 Courier Staffing and Capacity Management in On-Demand Delivery

Our problem addresses dynamic courier capacity acquisition in food delivery platforms, where the decision-maker must balance service reliability against operational costs by determining when to hire on-demand couriers. This setting differs fundamentally from classical vehicle routing, as the primary challenge lies in workforce capacity planning rather than route optimization.

Traditional approaches to workforce scheduling in service operations have relied on stochastic optimization frameworks. Bandi and Gupta (6) formulated operating room staffing as a robust optimization problem incorporating demand uncertainty through distributionally robust constraints. In the context of delivery operations, Ulmer and Savelsbergh (18) examined hybrid workforce systems combining scheduled employees with flexible crowdsourced labor through a two-stage stochastic program. Luy et al. (19) extended this line by addressing strategic workforce planning with hybrid driver fleets over extended planning horizons using fluid approximation models to balance reliable scheduled capacity against uncertain crowdsourced supply.

More recently, data-driven approaches have emerged to enable adaptive staffing decisions. Behrendt, Savelsbergh, and Wang (20) proposed sample average approximation combined with neural network surrogates to handle courier availability uncertainty in dispatch planning. Auad et al. (21) advanced this direction by applying deep Q-learning to dynamic courier hiring, where the agent learns a policy that directly maps system states to hiring actions.

Saleh et al. (22) proposed a DQN-based approach for dynamically extending committed couriers' shifts in response to real-time demand fluctuations. Building on this work, Saleh et al. (23) compared multiple RL algorithms (DQN, PPO, A2C) for online courier scheduling, demonstrating that PPO achieved superior performance through its clipped objective function.

2.2 Optimization Methods and RL for Scheduling

Exact optimization methods formulate workforce scheduling as mathematical programming problems to obtain provably optimal solutions. Beliën et al. (24) developed a mixed-integer linear programming model for aircraft maintenance workforce scheduling that generates cost-efficient schedules through column generation techniques. Alfares (25) proposed branch-and-bound algorithms for employee days-off scheduling that enumerate feasible patterns while pruning infeasible branches. Ernst et al. (26) presented integer programming formulations for nurse rostering problems that incorporate shift coverage constraints and work regulations. Atlason, Epelman, and Henderson (27) introduced an analytic center cutting plane method combined with simulation for call center staffing that iteratively refines staffing levels through constraint generation.

Heuristic methods provide computationally efficient approximate solutions for large-scale routing and scheduling problems. Gendreau, Hertz, and Laporte (28) developed a tabu search heuristic for vehicle routing with time windows that maintains solution feasibility through specialized neighborhood structures. Ropke and Pisinger (29) introduced adaptive large neighborhood search (ALNS) for pickup-and-delivery problems, which dynamically selects among multiple destroy-and-repair operators based on historical performance. Cordeau, Laporte, and Mercier (30) proposed a unified tabu search framework applicable to multiple vehicle routing variants through flexible constraint handling mechanisms. Vidal et al. (31) designed a hybrid genetic search combining population-based exploration with local search intensification, demonstrating state-of-the-art performance on capacitated vehicle routing benchmarks.

RL provides a natural framework for sequential decision-making under uncertainty, where agents learn value functions that guide action selection through experience. Van Hasselt, Guez, and Silver (32) mitigated overestimation bias through Double DQN's decoupled target computation, while Hessel et al. (33) integrated multiple algorithmic improvements—including distributional learning—into a unified architecture demonstrating superior sample efficiency on Atari benchmarks.

Distributional reinforcement learning extends this foundation by modeling the full random return $Z(s, a)$ rather than its expectation. Bellemare, Dabney, and Munos (34) introduced categorical distributional RL (C51), which discretizes the return support into fixed atoms and learns their probabilities. Dabney et al. (35) proposed Quantile Regression DQN (QR-DQN), which directly estimates quantiles of the return distribution at predetermined levels through asymmetric quantile regression losses. Extending this approach, Dabney et al. (36) developed IQN, which parameterize the continuous quantile function through neural embeddings, allowing agents to sample arbitrary risk levels and optimize beyond risk-neutral expectations by targeting conditional value-at-risk (CVaR). Mao et al. (37) applied distributional RL to

cluster scheduling, demonstrating that modeling job completion time distributions enables policies that maintain quality-of-service guarantees under load variability.

Architectural decomposition via MoE provides complementary capabilities by enabling specialized sub-networks to handle distinct input regimes. Jacobs et al. (38) formalized the adaptive expert framework, where a gating network routes inputs to domain-specialized modules based on learned relevance scores. Shazeer et al. (39) scaled this paradigm to massive networks through sparse gating, demonstrating that conditional computation can improve model capacity without proportional computational cost. In multi-task RL, Devin et al. (40) showed that modular networks facilitate transfer across robotic manipulation tasks, while Yang et al. (41) proposed soft expert routing mechanisms that allow gradient-based learning of task decompositions.

This paper integrates distributional RL with MoE architectures to achieve regime-adaptive value estimation for delivery operations, where operational regimes shift continuously and require dynamic expert reweighting based on real-time system state.

3 Problem Formulation

We model the dynamic courier scheduling problem as a Markov Decision Process (MDP) $\langle \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the state transition probability, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\gamma \in [0, 1)$ is the discount factor. The agent's objective is to learn a policy $\pi : \mathcal{S} \to \mathcal{A}$ that maximizes the expected cumulative discounted reward:

$$J(\pi) = \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{T} \gamma^t r_t \right] \tag{1}$$

where $\tau = (s_0, a_0, r_0, s_1, \ldots)$ denotes a trajectory sampled from policy π, and T corresponds to the end of the operational period ($H_0 = 450$ minutes).

State Space The state $s_t \in \mathcal{S} \subseteq \mathbb{R}^7$ at decision epoch t captures both order dynamics and courier availability. Each feature is normalized to $[0, 1]$:

$$s_t = [s_t^{(1)}, s_t^{(2)}, s_t^{(3)}, s_t^{(4)}, s_t^{(5)}, s_t^{(6)}, s_t^{(7)}] \tag{2}$$

where:

- $s_t^{(1)} = \frac{|\mathcal{O}_t^u|}{100}$: Number of unassigned orders
- $s_t^{(2)} = \frac{1}{60} \cdot \frac{1}{|\mathcal{O}_t^u|} \sum_{o \in \mathcal{O}_t^u} (d_o - t)$: Mean remaining time until deadline
- $s_t^{(3)} = \frac{1}{60} \cdot \min_{o \in \mathcal{O}_t^u} (d_o - t)$: Minimum remaining time, capturing urgency
- $s_t^{(4)} = \frac{1}{60} \cdot \frac{1}{|\mathcal{O}_t^u|} \sum_{o \in \mathcal{O}_t^u} (t - \tau_o)$: Mean waiting time for unassigned orders

- $s_t^{(5)} = \frac{|\{o \in \mathcal{O}_t : t-30 < \tau_o \leq t\}|}{20}$: Recent order arrivals (past 30 minutes)
- $s_t^{(6)} = \frac{|\mathcal{C}_t^a|}{50}$: Number of currently active couriers
- $s_t^{(7)} = \frac{1}{120} \cdot \frac{1}{|\mathcal{C}_t^a|} \sum_{c \in \mathcal{C}_t^a} (t_c^{\text{end}} - t)$: Mean remaining working time for active couriers

The normalization constants (100, 60, 50, 120) are derived from empirical maximums observed during preliminary training. This representation jointly captures order pressure (features 1–5) and resource availability (features 6–7).

Courier Hiring Actions (a_1 to a_5): For actions $(n_1, n_{1.5})$, the agent hires n_1 couriers with 1-hour shifts and $n_{1.5}$ couriers with 1.5-hour shifts. New couriers arrive after delay $\delta = 5$ minutes, incurring costs $K_c^{(1)} = -0.2$ and $K_c^{(1.5)} = -0.25$ respectively.

Reward Function The reward function balances service quality and operational costs:

$$r_t = r_t^{\text{lost}} + r_t^{\text{courier}} \tag{3}$$

An order is lost if it cannot meet its deadline even if immediately assigned. The penalty is:

$$r_t^{\text{lost}} = K_{\text{lost}} \cdot \Delta L_t \tag{4}$$

where $K_{\text{lost}} = -1$ and $\Delta L_t = L_t - L_{t-1}$ is the number of newly lost orders.

The r_t^{courier} is defined as follows:

$$r_t^{\text{courier}} = \begin{cases} K_c^{(1)} \cdot n_1 + K_c^{(1.5)} \cdot n_{1.5}, & \text{if } a_t \in \{a_1, \ldots, a_5\} \\ K_{\text{extend}} \cdot |\mathcal{C}_t^{\text{ext}}|, & \text{if } a_t = a_6 \end{cases} \tag{5}$$

We use raw rewards without normalization to preserve cost trade-offs, with magnitudes naturally falling within $[-2.0, 0]$ per step.

Transition Dynamics The transition probability $P(s_{t+1}|s_t, a_t)$ is governed by:

- **Order arrivals:** Time-varying Poisson process with peak periods (11:30–13:00, 18:00–19:30)
- **Courier scheduling:** Hired couriers start at $t + \delta$; extended couriers gain 30 minutes immediately
- **Order assignment:** Greedy assignment in ascending deadline order, subject to feasibility constraints

4 Methodology

Due to the dynamic and stochastic nature of real-time order arrivals and courier availability, dynamic decision-making is crucial for solving the Dynamic Courier Capacity Acquisition problem. To handle the sequential decision process under uncertainty, we introduce a DRL framework based on IQN, regarding the courier dispatching problem as a Markov Decision Process (MDP), which enables the agent to learn risk-sensitive scheduling policies by modeling the full return distribution rather than merely its expectation. However, different operating scenarios, such as traffic surges during peak hours and off-peak hours, often lead to unstable quality of solutions in single-network deep reinforcement learning methods. To further improve the solution quality, we propose an integrated architecture integrating the our framework with a MoE encoder, where multiple expert sub-networks specialize in different demand regimes, and a gating mechanism dynamically selects the most appropriate experts based on the current system state. Furthermore, we design a novel shift-extension mechanism that allows the agent to extend the working hours of couriers whose shifts are about to end., complementing the conventional on-demand courier dispatching strategy and providing a more cost-effective response to short-term demand spikes.

4.1 Overall Framework

As illustrated in Figure 1, we formulate the dynamic courier capacity acquisition problem as a MDP and solve it with a DRL framework. The system operates in a feedback loop between the agent and the courier delivery environment. At each time step, the agent observes the state s_t, which is processed by a Distributional Q-Network to estimate the action-value distribution. Based on this estimation, the agent selects an action a_t and interacts with the environment. The environment then returns the next state s_{t+1} along with a reward r_t, which is used to update the agent's network parameters. This process is repeated over time to improve the decision policy.

We now detail the state representation, action space, and decision process at each decision epoch. At each decision epoch t, the environment computes the current system state s_t, a compact vector that summarizes the supply–demand situation of the delivery system, including the number of unassigned orders, order deadlines, recent demand trends, and available courier capacity. The state s_t is fed into the Distributional Q-Network, as illustrated in Figure 1. . The network evaluates every candidate action and outputs an action-value distribution for each action. The agent then selects an action a_t using a ϵ-greedy strategy: with probability $1 - \epsilon$ it chooses the action with the best estimated value and with probability ϵ it selects a random action to explore the action space.

The action space consists of two types of actions: *Hire Couriers*, which hires additional on-demand couriers, and *Extend Shift*, which extends the working hours of couriers approaching the end of their shifts. At each time step t, the environment first identifies all unassigned orders and sequentially assigns them to available couriers, then updates the states of both orders and couriers accordingly. The environment returns a scalar reward r_t, which combines a penalty for newly lost orders and the cost of hiring or extending couriers, along with the next state s_{t+1}. The transition (s_t, a_t, r_t, s_{t+1}) is stored in an experience replay buffer for

training.

Periodically, a mini-batch of past transitions is sampled from the replay buffer. The network's predicted action-value distributions are compared against target values computed from a separate, slowly updated target network. The discrepancy is measured by a distributional loss, which is introduced in Section 4.2, and the network parameters are updated via gradient descent. This off-policy learning scheme allows the agent to learn from historical experience without requiring new interactions for every update. At the end of each training epoch, the learned policy is evaluated on a set of independent test episodes.

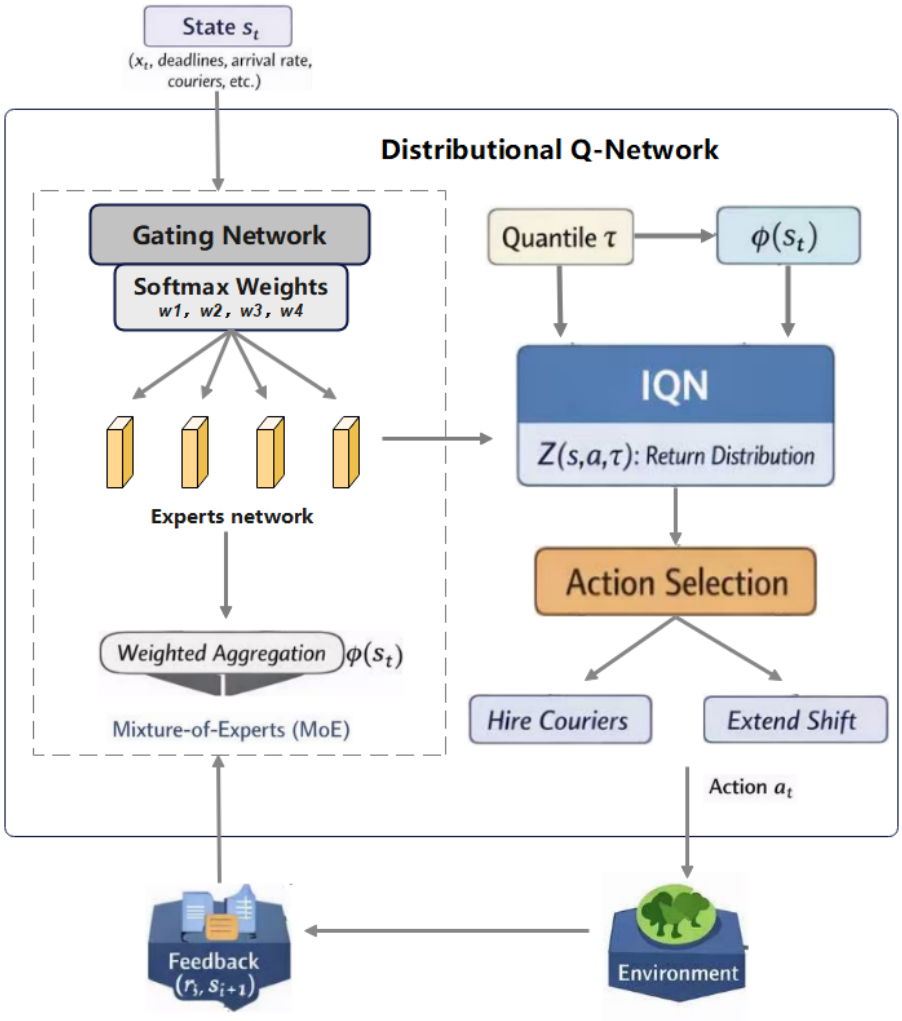

Figure 1: Overall Algorithm Architecture

Proactive Shift Extension for Zero-Latency Capacity Control Traditional delivery scheduling typically relies on reactive courier hiring, whose effect is delayed by onboarding period(i.e., the time required for newly hired couriers to become operational). Such action latency can cause short-term capacity shortfalls during demand surges. To mitigate this issue,

we augment the action space with a zero-latency capacity control option that immediately increases effective supply by extending the shifts of currently active couriers. The resulting discrete action space $\mathcal{A}$ contains 7 actions:

$$\mathcal{A} = \{(n_1, n_{1.5})\}_{(n_1,n_{1.5})\in\{0,1,2\}^2} \cup \{\text{EXTEND}\}. \tag{6}$$

Table 1: Action Space Definition

Index	Action	Cost
a_0	$(0,0)$ No action	0
a_1	$(1,0)$ Add 1 courier (1-hour shift)	-0.2
a_2	$(0,1)$ Add 1 courier (1.5-hour shift)	-0.25
a_3	$(2,0)$ Add 2 couriers (1-hour shifts)	-0.4
a_4	$(1,1)$ Add 1+1 couriers (mixed shifts)	-0.45
a_5	$(0,2)$ Add 2 couriers (1.5-hour shifts)	-0.5
a_6	EXTEND (extend shifts by 30 min)	$-0.15n$

EXTEND action is an additional discrete action available to the agent alongside the courier hiring actions. When selected, the environment executes the following shift-extension procedure. First, the system identifies all couriers whose shift is approaching its scheduled end:

$$\mathcal{C}_t^{\text{ext}} = \{c \in \mathcal{C}_t^a : t < t_c^{\text{end}} \le t + \Delta_{\text{ext}}\}, \tag{7}$$

where $\mathcal{C}_t^a$ denotes the set of currently active couriers at decision epoch t, t_c^{end} is the scheduled shift end time of courier c, and Δ_{ext} is the look-ahead window. For each courier $c \in \mathcal{C}_t^{\text{ext}}$, the shift is extended by a fixed duration:

$$t_c^{\text{end}} \leftarrow t_c^{\text{end}} + \Delta_{\text{ext}}, \tag{8}$$

incurring a per-courier cost of K_{extend}, which is lower than the cost of hiring a new on-demand courier. If no couriers are approaching the end of their shift, the action has no effect and incurs no cost. The hiring of new couriers incurs an onboarding delay δ before they become available. In contrast, the extension of the shift provides an immediate and cost-effective response to short-term demand surges by retaining current active couriers.

4.2 Distributional Q-Network Design

The Distributional Q-Network serves as the core component of the proposed framework and maps the system state to risk-sensitive action values. It consists of two modules: an MoE state encoder that produces a regime adaptive representation, and an IQN head that estimates the full return distribution.

4.2.1 MoE State Encoder

In rapid delivery systems, state observations are highly heterogeneous due to fluctuations in demand and courier availability. A single monolithic encoder may be insufficient to represent such highly variable dynamics with one shared set of parameters. Inspired by the MoE

paradigm (39; 38), we replace the conventional MLP state encoder with an MoE module. A learned gating network then adaptively combines specialized expert subnetworks to produce a regime aware state representation.

The MoE state encoder consists of two components: a set of K expert networks and a gating network. Each expert $e \in \{1, \ldots, K\}$ is an independent multi-layer perceptron that maps the input state vector $s_t \in^d$ (d=7)to a feature representation:

$$f_e(s_t) = \mathrm{MLP}_e(s_t) \in \mathbb{R}^n, \quad e = 1, \ldots, K. \tag{9}$$

All experts share the same architecture in terms of layer sizes and activation functions, but use independent parameters, so that each expert can specialize in encoding different operational patterns.

The gating network $g(s_t)$ takes the same state s_t as input and produces a probability distribution over the experts:

$$g(s_t) = \mathrm{Softmax}\big(W_2\ \mathrm{ReLU}(W_1\, s_t + b_1) + b_2\big) \in \mathbb{R}^K, \tag{10}$$

where W_1, b_1, W_2, b_2 are learnable parameters. The e-th component $g_e(s_t)$ represents the relevance of expert e to the current state. The final state feature $\varphi(s_t)$ is computed as a convex combination of all expert outputs:

$$\varphi(s_t) = \sum_{e=1}^{K} g_e(s_t) \cdot f_e(s_t). \tag{11}$$

This soft routing mechanism allows the encoder to smoothly interpolate between different expert representations, rather than performing hard routing that might introduce non-differentiable switching. In practice, the gating network learns to assign higher weights to experts that have specialized in the current operational regime. For example, one expert may become attuned to peak-hour states with high demand, while another focuses on off-peak states with surplus capacity.

A known issue with MoE architectures is *expert collapse*, where the gating network converges to routing all inputs to a single expert, leaving the remaining experts untrained. To mitigate this issue, we follow prior work (39) on MoE training and add an auxiliary load-balancing loss to encourage uniform expert utilization across each training batch. Let $\bar{g}_e = \frac{1}{B} \sum_{i=1}^{B} g_e(s_t^{(i)})$ denote the average gating weight of expert e over a mini-batch of size B. The load-balancing loss is defined as:

$$\mathcal{L}_{\text{balance}} = \mathrm{MSE}\left(\bar{g},\ \frac{1}{K}\mathbf{1}\right) = \frac{1}{K} \sum_{e=1}^{K} \left(\bar{g}_e - \frac{1}{K}\right)^2, \tag{12}$$

The MoE state encoder produces the feature vector $\varphi(s_t)$, which is then passed to the IQN module for quantile-conditioned value estimation.

4.2.2 Distributional value estimation

Standard value-based RL methods estimate the expected action value $Q(s,a) = [Z(s,a)]$, where $Z(s,a)$ denotes the random return. While this scalar expectation suffices for risk-neutral decision-making, it discards distributional information that is critical in logistics operations where worst-case scenarios (e.g., massive order losses during peak hours) must be explicitly managed. IQN (36) address this limitation by learning the full quantile function of the return distribution.

Instead of a fixed set of quantile levels, IQN treats the quantile fraction $\tau \in (0,1)$ as a continuous input and learns a mapping

$$Z_\theta(s,a,\tau) \approx F^{-1}_{Z(s,a)}(\tau), \tag{13}$$

where $F^{-1}_{Z(s,a)}$ is the inverse cumulative distribution function (quantile function) of the return. For any sampled τ, the network output $Z_\theta(s,a,\tau)$ represents the τ-quantile of the return distribution under state s and action a. This implicit parameterization allows the network to estimate an arbitrary number of quantiles without being constrained to a predefined grid.

To incorporate the quantile fraction τ into the network, IQN employs a cosine basis embedding. Given a sampled τ, the embedding vector $\psi(\tau) \in^n$ is computed as:

$$\psi_j(\tau) = \mathrm{ReLU}\left(\sum_{i=0}^{N-1} \cos(\pi\, i\, \tau)\cdot w_{ij} + b_j\right), \quad j = 1,\dots,n, \tag{14}$$

where N is the number of cosine basis functions, and w_{ij}, b_j are learnable parameters of a fully-connected layer. This embedding maps the scalar τ into a high-dimensional feature space that captures periodic structure across quantile levels, enabling the network to distinguish between different parts of the return distribution (e.g., the left tail corresponding to pessimistic outcomes versus the right tail corresponding to optimistic ones).

The state observation s_t is first processed by a state encoder to produce a feature vector $\varphi(s_t) \in \mathbb{R}^n$. The state feature and the quantile embedding are then fused via element-wise multiplication:

$$h(s_t,\tau) = \varphi(s_t) \odot \psi(\tau), \tag{15}$$

where $\odot$ denotes the Hadamard (element-wise) product. The fused representation $h(s_t,\tau)$ is subsequently passed through a merge network—a fully-connected layer followed by ReLU activation—and a final linear head to produce the quantile value for each action:

$$Z_\theta(s_t,a,\tau) = W_q\ \mathrm{ReLU}(W_m\, h(s_t,\tau) + b_m) + b_q, \tag{16}$$

where W_m, b_m, W_q, b_q are learnable parameters of the merge network and Q-value head, respectively.

During each training step, a set of quantile fractions $\{\tau_k\}_{k=1}^{K}$ and $\{\tau'_k\}_{k=1}^{K'}$ are independently sampled from $\mathcal{U}(0,1)$. The network is trained by minimizing the quantile Huber loss:

$$\mathcal{L}_{\mathrm{IQN}} = \frac{1}{KK'}\sum_{k=1}^{K}\sum_{k'=1}^{K'} \rho^{\kappa}_{\tau_k}\big(\hat{T}Z_{\theta^-}(s',a^*,\tau'_{k'}) - Z_\theta(s,a,\tau_k)\big), \tag{17}$$

where $\hat{T}Z_{\theta^-}$ denotes the distributional Bellman target computed with the target network θ^-, $a^* =_a \tau'[Z_{\theta^-}(s', a, \tau')]$ is the greedy next action, and ρ_τ^κ is the quantile Huber loss defined as:

$$\begin{aligned} \rho_\tau^\kappa(\delta) &= |\tau - \mathbf{1}\{\delta < 0\}| \cdot \mathcal{L}_\kappa(\delta), \\ \mathcal{L}_\kappa(\delta) &= \begin{cases} \frac{1}{2}\delta^2/\kappa, & \text{if } |\delta| \leq \kappa, \\ |\delta| - \frac{1}{2}\kappa, & \text{otherwise.} \end{cases} \end{aligned} \tag{18}$$

This asymmetric loss ensures that underestimation and overestimation are penalized differently depending on the quantile level τ, which drives the network to accurately approximate the entire return distribution.

5 Experiments

5.1 Experimental Setup

Our experiments are conducted in a simulation environment calibrated with real-world delivery data. The operational cycle is set to $H_0 = 450$ minutes, corresponding to approximately 89 decision steps per episode. To accelerate data collection, we utilize 10 parallel environments for simultaneous trajectory sampling. The network architectures are configured as follows.

- **IQN Architecture:** The quantile embedding layer utilizes $n = 64$ cosine basis functions with an embedding dimension of 256. The feature fusion layer combines state and quantile embeddings via element-wise multiplication, followed by a dense layer outputting the action-value distribution.
- **MoE Architecture:** The backbone consists of $K = 4$ expert networks, where each expert is an independent Multi-Layer Perceptron (MLP) with hidden layers of sizes $[512, 512, 256]$. The Gating Network employs a lightweight structure with a hidden layer of 64 units to output Softmax-normalized routing weights.

All models are optimized using the Adam optimizer. We detail the specific hyperparameter settings for the IQN-MoE framework and training process in Table 2. These values were determined through grid search to balance convergence speed and stability. Specifically, the load balance coefficient λ_{bal} is set to 0.01 to prevent expert collapse without dominating the primary Q-learning objective.

5.2 Satisfaction Metric

Satisfaction Metric To evaluate service quality from the user perspective, we define a *Customer Satisfaction* metric that translates the lost order rate into a quality score reflecting user experience. Unlike linear metrics that fail to capture the psychological impact of service degradation, we employ a piecewise satisfaction model that segments user tolerance across different failure regimes:

Table 2: IQN-MoE Network and Training Parameters

Parameter	**Value**
Hidden layers	[512, 512, 256]
Number of experts K	4
Cosine bases N	64
Load balance coefficient λ	0.01
Learning rate	5×10^{-5}
Discount factor γ	0.99
Batch size	256
Buffer size	20,000
Training quantiles K	32
Online quantiles K'	8
Target update frequency	500
N-step returns	3
ϵ-greedy (train/test)	0.1 / 0.05
Total training steps	500,000

$$\text{Satisfaction}(r) = \begin{cases} 100 - 0.8r, & \text{if } r < 10\% \\ 92 - 1.0(r - 10), & \text{if } 10\% \leq r < 30\% \\ 72 - 0.8(r - 30), & \text{if } 30\% \leq r < 60\% \\ 48 - 0.6(r - 60), & \text{if } r \geq 60\% \end{cases} \tag{19}$$

where r denotes the lost order rate (percentage of orders that cannot meet their deadlines). This piecewise function is designed based on the following rationale:

- **Region 1** ($r < 10\%$): Excellent service zone with minimal penalty (coefficient 0.8). Low failure rates are nearly imperceptible to users in on-demand delivery contexts.
- **Region 2** ($10\% \leq r < 30\%$): Good service zone with linear degradation (coefficient 1.0). Users begin perceiving service quality issues but remain generally satisfied.
- **Region 3** ($30\% \leq r < 60\%$): Acceptable service zone with moderate penalty (coefficient 0.8). Users experience noticeable dissatisfaction but the system remains operational.
- **Region 4** ($r \geq 60\%$): Poor service zone with reduced penalty (coefficient 0.6). This gentler slope prevents premature saturation at zero, maintaining discriminative power between severely degraded policies.

The piecewise coefficients ensure continuity at segment boundaries while providing differentiated sensitivity across operational regimes. For instance, at $r = 4.7\%$ (typical for well-performing policies at small scale), the satisfaction is $100 - 0.8 \times 4.7 = 96.24\%$, whereas

at $r = 53.5\%$ (common at large scale under resource constraints), satisfaction drops to $72 - 0.8 \times (53.5 - 30) = 53.20\%$, appropriately reflecting the degraded but non-catastrophic service state.

This metric complements the original reward signal and provides an interpretable measure of quality that is consistent with user experience.

5.3 Baseline Policies

To contextualize the performance of the proposed method, we compare against the following baselines:

Random Policy: At each time step, actions are sampled uniformly from the discrete action set.

OPC baseline We additionally consider a rule-based heuristic policy that makes staffing decisions based on the Order-Per-Courier (OPC) metric. Specifically, the policy computes:

$$\mathrm{OPC}(t) = \frac{N_{\text{unassigned}}(t)}{N_{\text{available}}(t)}, \tag{20}$$

which quantifies the workload pressure at decision epoch t. When the current OPC exceeds a threshold $\mathrm{OPC}_{\max}$, the policy determines the number of couriers to hire as:

$$m_t = \left\lceil \frac{N_{\text{unassigned}}(t)}{\mathrm{OPC}_{\text{target}}} \right\rceil - N_{\text{available}}(t), \tag{21}$$

where $\mathrm{OPC}_{\text{target}}$ represents the desired workload level per courier. We set $\mathrm{OPC}_{\max} = 1.5$ and $\mathrm{OPC}_{\text{target}} = 1.0$ in our experiments.

Based on this formulation, we evaluate three policy variants:

- $\boldsymbol{\pi^1}$ **(1-hour only)**: Exclusively hires 1-hour couriers to meet the computed demand m_t.
- $\boldsymbol{\pi^{1.5}}$ **(1.5-hour only)**: Exclusively hires 1.5-hour couriers to meet the computed demand m_t.
- $\boldsymbol{\pi}^{\text{hybrid}}$ **(Time-dependent)**: Dynamically selects courier type based on the current time period. During peak hours (lunch: $t \in [60, 120]$ min; dinner: $t \in [300, 360]$ min), the policy hires 1-hour couriers for rapid response; otherwise, it hires 1.5-hour couriers for cost efficiency.

If $\mathrm{OPC}(t) \leq \mathrm{OPC}_{\max}$, no hiring action is taken ($m_t = 0$).

Heuristic Policy: We additionally consider a rule-based heuristic policy that makes staffing decisions based on three explicit state indicators. Specifically, the policy uses:

(i) *Demand Pressure*, $\mathrm{DP} = \frac{N_{\text{unassigned}}}{N_{\text{available}}+1}$, to quantify the imbalance between demand and available capacity;

(ii) an *Urgency* flag, which is activated when any unassigned order is within 15 minutes of its deadline;

(iii) a *Growth* indicator, which signals a recent demand surge of more than 10 new orders in the past 30 minutes.

Based on these indicators, the heuristic follows a tiered response strategy:

- **Extreme Pressure** ($\mathrm{DP} > 4$): If the active courier pool is small (< 10), the policy hires two 1.5-hour couriers; otherwise it hires one 1-hour and one 1.5-hour courier.
- **High Pressure** ($2 < \mathrm{DP} \leq 4$): If *Urgency* is detected, the policy hires one 1.5-hour courier; otherwise, it randomly hires either one or two 1-hour couriers.
- **Medium Pressure** ($1 < \mathrm{DP} \leq 2$): If *Urgency* or *Growth* is detected, the policy hires one 1-hour courier; otherwise, no hiring is performed.
- **Low Pressure** ($\mathrm{DP} \leq 1$): No hiring is performed.

DQN Baseline (Value-Based): Following the work of Auad et al. (42), we adopt a standard Deep Q-Network that predicts scalar action-values as a baseline. The update target follows the deterministic Bellman backup

$$y = r + \gamma \max_{a'} Q_{\theta^-}(s', a'), \tag{22}$$

and the network parameters are fitted to this target in expectation. This baseline reflects the conventional approach of estimating only the mean return, without modeling distributional variability.

Training Dynamics and Convergence Analysis

To validate the efficiency of distributional modeling, we visualize training dynamics across 500 episodes at the 200-order scale (Figure 2). Learning-based methods (IQN, DQN) are evaluated every epoch using 5 test episodes, while baselines (Heuristic, Random) represent fixed-performance references.

Convergence Speed. IQN reaches stability at **epoch 12**, whereas DQN requires **epoch 15**—a 20% reduction in training time. This acceleration stems from IQN's distributional Bellman operator, which provides richer gradient signals by modeling the full return distribution rather than scalar expectations.

Final Performance. Measured over the final 10 epochs, IQN achieves -37.32 ± 4.16, outperforming DQN (-39.56 ± 3.59) by 5.7%, Random (-58.06 ± 0.91) by 35.7%, and Heuristic (-61.72 ± 0.55) by 39.5%. The performance hierarchy validates that learned policies substantially outperform static rules, while distributional modeling provides consistent advantages over standard Q-learning.

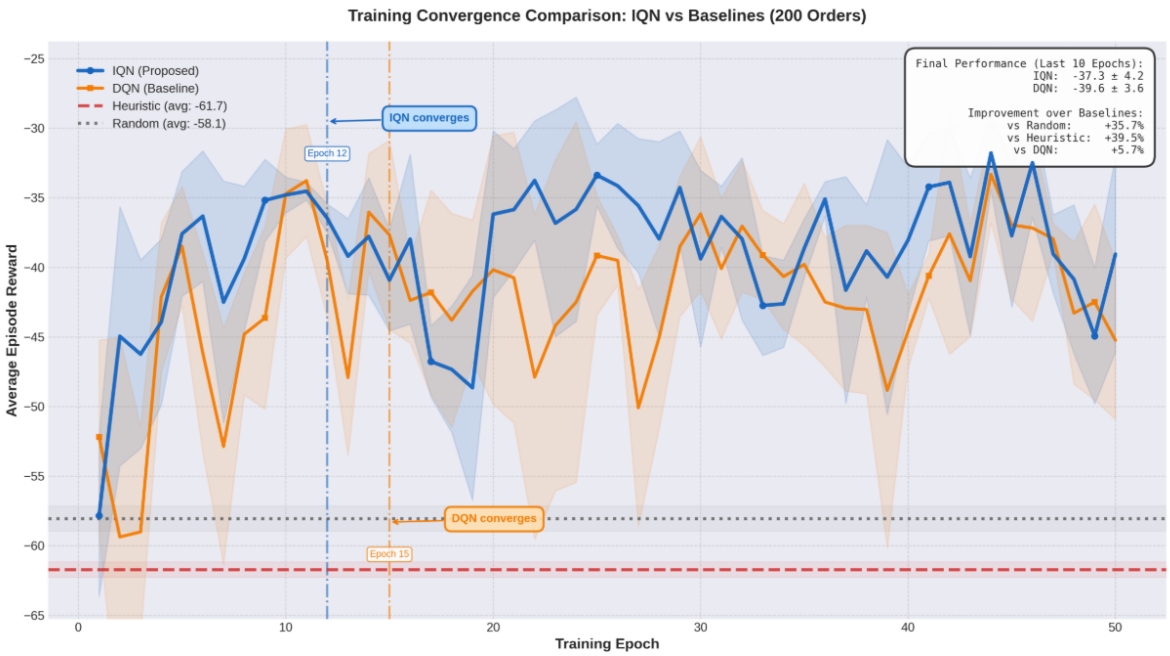

Figure 2: Training convergence comparison at 200-order scale. IQN (blue) converges at epoch 12, 20% faster than DQN (orange, epoch 15). Shaded regions indicate ±1 standard deviation. Vertical markers denote convergence points where reward variance stabilizes.

Training Stability. Both learning methods exhibit controlled variance after convergence (IQN: $\sigma = 4.16$; DQN: $\sigma = 3.59$), ensuring reproducible performance across training runs. The modest increase in variance compared to fixed baselines ($\sigma < 1.0$) is vastly outweighed by the 35–40% performance gains, confirming the practical utility of learning-based approaches.

5.4 Main Results

We conduct three groups of experiments to systematically evaluate our framework. Table 3 summarizes the comprehensive performance metrics across three order scales (200, 500, and 1000 orders). We organize the results according to three experimental groups: algorithm comparison (Exp 1), MoE expert ablation (Exp 2), and Extend-Shift mechanism ablation (Exp 3).

5.4.1 Experiment 1: Algorithm Comparison

We compare the proposed IQN framework against Random, Heuristic, OPC baseline, and DQN across three operational scales. Figure 3 visualizes the reward comparison.

200 Orders. IQN achieves the best performance with average reward −38.9 (+33.0% over Random) and 96.24% satisfaction (+11.00pp over Random). DQN also performs well with reward −40.5 (+30.3%) and 95.12% satisfaction (+9.88pp). Both RL methods substantially outperform rule-based approaches: Heuristic achieves only −61.7 reward (−6.2%) with

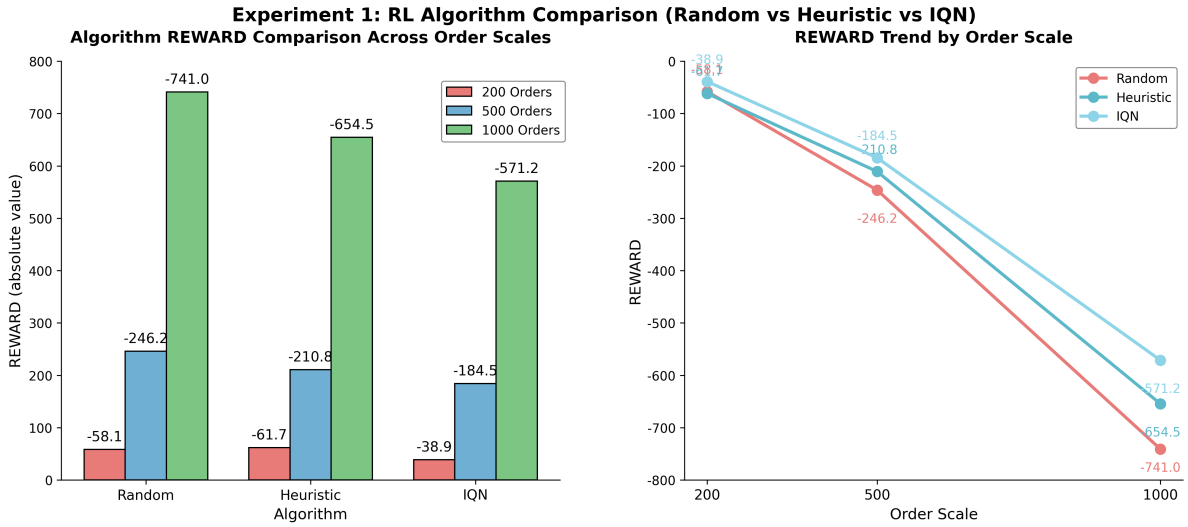

Figure 3: Algorithm comparison across order scales. IQN consistently achieves the highest rewards across all scales.

83.77% satisfaction (−1.47pp), actually underperforming Random (−58.1 reward, 85.24% satisfaction).

500 Orders. As complexity increases, the performance gap widens. IQN achieves −184.5 reward (+25.0%) and 73.00% satisfaction (+11.64pp), while DQN reaches −188.7 (+23.3%) and 72.08% (+10.72pp). Heuristic improves to −210.7 (+14.4%) with 67.71% satisfaction (+6.35pp), but remains significantly behind RL methods. Both IQN and DQN invest substantially more in courier costs (IQN: 39.6, +48.3%; DQN: 39.1, +46.4%) compared to Random (26.7).

1000 Orders. Under severe resource constraints, Random collapses to −741.0 reward and 41.82% satisfaction. IQN maintains −571.2 reward (+22.9%) and 53.20% satisfaction (+11.38pp), while DQN achieves −573.4 (+22.6%) and 53.00% (+11.18pp). The marginal difference between IQN and DQN (+0.20pp) indicates that both methods adopt similarly aggressive strategies under system overload.

5.4.2 Experiment 2: MoE Expert Ablation

We evaluate the impact of expert count $K \in \{2, 4, 6\}$ on performance. Figure 4 illustrates the results.

200 Orders. Expert count has negligible impact: MoE-2 achieves −39.4 reward and 95.52% satisfaction, MoE-4 achieves −40.0 (−1.5%) and 95.36% (−0.16pp), and MoE-6 achieves −38.3 (+2.8%) and 96.08% (+0.56pp). The performance gap between configurations is minimal (<1pp in satisfaction).

500 Orders. A clear hierarchy emerges. MoE-4 achieves the best performance with −178.4 reward (+8.1% over MoE-2) and 74.20% satisfaction (+3.32pp). Notably, MoE-6 (−183.1, +5.7%) underperforms MoE-4, achieving only 73.10% satisfaction (+2.22pp), suggesting that excessive experts introduce routing noise at this scale.

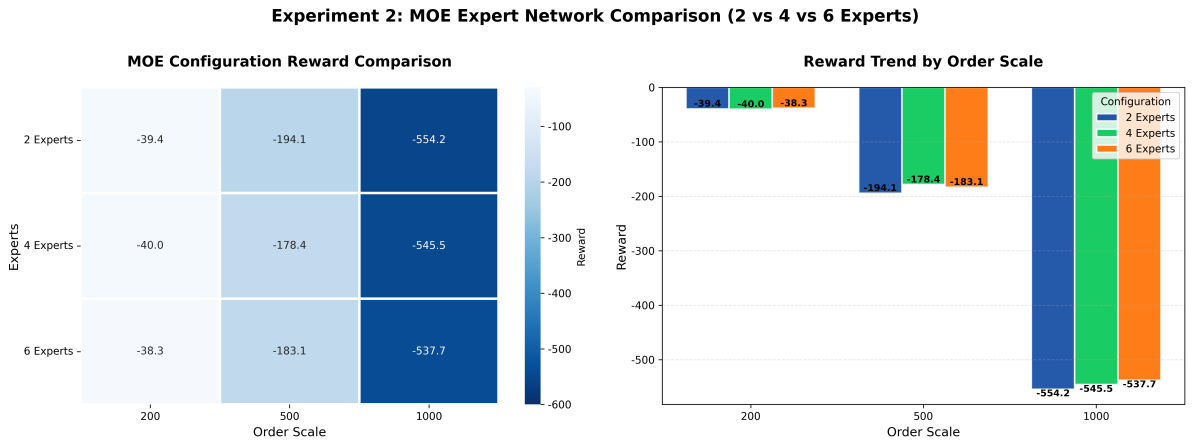

Figure 4: Impact of expert count on reward across scales. $K = 4$ provides optimal trade-off at medium scales.

1000 Orders. Higher complexity benefits from more experts. MoE-6 achieves the best reward -537.7 ($+3.0\%$) and satisfaction 56.08% ($+1.44$pp), followed by MoE-4 (-545.5, $+1.6\%$, 55.44%) and MoE-2 (-554.2, 54.64%). However, the gains from MoE-6 over MoE-4 are marginal ($+0.64$pp), while MoE-4 reduces computational overhead by 33%.

Configuration Selection. Based on these results, we adopt $K = 4$ as the default configuration, offering the optimal balance between performance and computational efficiency across scales.

5.4.3 Experiment 3: Extend-Shift Mechanism Ablation

We evaluate the Extend-Shift mechanism with three extension durations (30, 60, 90 minutes) and compare against DQN with the same mechanism. Figure 5 shows the results.

200 Orders. Extend 30-min achieves -39.2 reward and 98.24% satisfaction. Longer extensions (60-min and 90-min) both achieve -37.4 reward ($+4.6\%$), but with slightly lower satisfaction (96.40% and 96.16%, respectively). DQN with Extend-Shift achieves -40.7 (-3.3%) and 95.12% (-3.12pp), confirming IQN's advantage even with action augmentation.

500 Orders. The Extend-Shift mechanism demonstrates transformative impact. Extend 30-min achieves -151.9 reward and 82.20% satisfaction—a substantial improvement over Exp 2's best (MoE-4: -178.4, 74.20%). Extend 90-min further improves to -120.1 ($+20.9\%$) and 88.50% ($+6.30$pp), while simultaneously *reducing* courier costs by 0.6% (52.7 vs 53.0). DQN achieves -127.0 ($+16.4\%$) but only 72.10% satisfaction (-10.10pp), indicating poor capacity allocation despite the extended action space.

1000 Orders. The mechanism's value is most pronounced at large scale. Extend 30-min achieves -479.0 reward and 62.72% satisfaction. Extend 90-min achieves the best overall performance: -313.3 ($+34.6\%$) and 78.50% ($+15.78$pp), with courier cost 78.4 ($+24.2\%$). DQN achieves -376.6 ($+21.4\%$) but only 52.96% satisfaction (-9.76pp), demonstrating that distributional modeling is essential for effective utilization of the Extend-Shift mechanism.

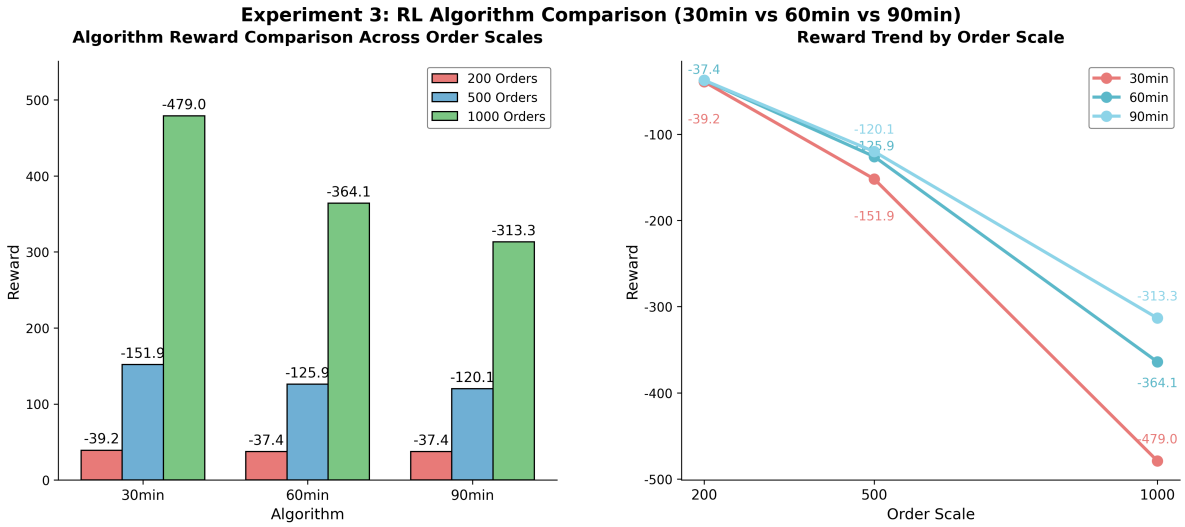

Figure 5: Ablation of Extend-Shift duration. Longer extensions yield substantial improvements at larger scales.

5.5 Discussion

In this section, we provide in-depth analysis of the experimental results, examining the underlying mechanisms that drive performance differences across methods and scales.

5.5.1 Why Does IQN Outperform DQN?

The consistent advantage of IQN over DQN (Exp 1) stems from distributional value estimation's ability to capture return uncertainty. At 200 orders, IQN's advantage is modest (+1.12pp satisfaction, +2.7% reward improvement over DQN) because the operational environment is relatively predictable. As scale increases to 500 orders, the advantage grows (+0.92pp, +1.7% reward) due to increased demand stochasticity. At 1000 orders, both methods converge to similar performance (+0.20pp, +0.3% reward) as the solution space narrows under severe resource constraints—both must adopt aggressive hiring regardless of risk modeling.

The cost-efficiency profiles reveal distinct strategies. At 500 orders, IQN invests slightly more in couriers (39.6 vs 39.1, +1.3%) but achieves disproportionately higher satisfaction gains (+11.64pp vs +10.72pp over Random). This 8.6% improvement in satisfaction gain per unit cost demonstrates that distributional modeling enables more precise capacity allocation, particularly during demand surges where tail-risk awareness prevents catastrophic underprovisioning.

5.5.2 The Pareto Frontier of Expert Capacity

The MoE ablation (Exp 2) reveals a non-monotonic relationship between expert count and performance. At 200 orders, all configurations achieve similar results because the operational

regimes are insufficiently diverse to benefit from specialized experts—the gating network cannot learn meaningful regime distinctions.

At 500 orders, MoE-4 achieves optimal performance (−178.4 reward, 74.20% satisfaction), outperforming both MoE-2 (+8.1% reward, +3.32pp satisfaction) and MoE-6 (+5.7%, +2.22pp).

At 1000 orders, MoE-6 marginally regains the lead (+3.0% over MoE-2, +1.44pp) as the increased data volume provides sufficient training signal for all experts. However, MoE-4 remains within 1.5% of optimal while reducing computational overhead by 33%, establishing it as the Pareto-optimal configuration.

5.5.3 The Transformative Power of Zero-Latency Capacity Control

Experiment 3 demonstrates that the Extend-Shift mechanism provides the most significant performance improvements, exceeding the gains from algorithmic enhancements (Exp 1) and architectural tuning (Exp 2).

Magnitude Analysis. Comparing across experiments at 1000 orders:

- Exp 1 (IQN vs Random): +22.9% reward improvement
- Exp 2 (MoE-6 vs MoE-2): +3.0% reward improvement
- Exp 3 (Extend 90-min vs Extend 30-min): +34.6% reward improvement

The Extend-Shift mechanism delivers 1.5× the improvement of algorithm selection and 11.5× the improvement of expert tuning, confirming that domain-aware action space design is the primary lever for operational breakthrough.

Cost-Efficiency Paradox. At 500 orders, Extend 90-min achieves a remarkable outcome: +6.30pp satisfaction improvement while *reducing* courier costs by 0.6% compared to Extend 30-min (52.7 vs 53.0). This "super-efficiency" arises because proactive shift extensions prevent the system from entering panic modes that trigger expensive reactive hiring. The agent learns to extend existing couriers preemptively, avoiding the 5-minute onboarding delay that causes order losses during demand spikes.

IQN vs DQN with Extend-Shift. The DQN results in Exp 3 reveal that distributional modeling is essential for effective mechanism utilization. Despite having access to the same Extend-Shift actions, DQN achieves substantially lower satisfaction:

- 500 orders: DQN 72.10% vs IQN 82.20% (Extend 30-min), −10.10pp gap
- 1000 orders: DQN 52.96% vs IQN 62.72% (Extend 30-min), −9.76pp gap

While DQN achieves competitive rewards (−127.0 vs −151.9 at 500 orders), it fails to translate this into service quality. This suggests that DQN's expectation-based optimization exploits the Extend-Shift mechanism for cost reduction rather than service improvement, whereas IQN's risk-aware learning correctly prioritizes satisfaction under uncertainty.

5.5.4 Scale-Dependent Performance Dynamics

The experimental results reveal distinct performance regimes across scales:

Small Scale (200 Orders): Saturation Regime. All methods achieve near-optimal performance (IQN: 96.24%, DQN: 95.12%). The operational complexity is insufficient to differentiate algorithmic sophistication. Expert count and extension duration have minimal impact, as the base capacity is adequate for demand.

Medium Scale (500 Orders): Differentiation Regime. Performance gaps emerge and widen. IQN's distributional advantage becomes meaningful (+0.92pp over DQN), MoE-4 significantly outperforms MoE-2 (+3.32pp), and Extend-Shift provides substantial gains (82.20% vs 74.20% for MoE-4 best). This scale represents the "sweet spot" where algorithmic and architectural choices materially impact outcomes.

Large Scale (1000 Orders): Constraint Regime. Resource scarcity dominates. The IQN-DQN gap narrows (+0.20pp) as both must adopt aggressive hiring. However, the Extend-Shift mechanism becomes critical: without it, even IQN achieves only 53.20% satisfaction (Exp 1); with 90-min extension, satisfaction jumps to 78.50% (Exp 3)—a 25.30pp improvement that dwarfs all other optimizations.

5.5.5 Practical Implications

These findings yield actionable insights for deployment:

1. **Prioritize Action Space Design**: The Extend-Shift mechanism delivers 5–10× larger improvements than algorithmic refinements. Practitioners should invest in domain-aware action augmentation before optimizing neural architectures.

2. **Scale-Appropriate Configuration**: Use $K = 4$ experts as default; increase to $K = 6$ only for very large-scale deployments (>1000 daily orders) where computational overhead is justified.

3. **Extension Duration Selection**: At small scales, 30-minute extensions suffice (98.24% satisfaction). At medium-to-large scales, 90-minute extensions provide substantial gains (+6.30pp at 500 orders, +15.78pp at 1000 orders) with favorable cost characteristics.

4. **Distributional Modeling is Essential**: DQN's failure to leverage Extend-Shift effectively (−10.10pp at 500 orders) demonstrates that risk-aware value estimation is necessary for complex action spaces—expectation-based methods may exploit mechanisms suboptimally.

5.6 Cost-Reward Efficiency Analysis

Contrary to the assumption that minimizing costs maximizes rewards, we observe a non-monotonic relationship where increased courier investment often yields net positive rewards. We quantify this trade-off using the marginal efficiency ratio $\eta = \Delta S/\Delta C$, where ΔS denotes satisfaction improvement (in percentage points) and ΔC denotes relative cost increase. Investment is profitable when higher costs translate to disproportionate satisfaction gains.

Table 3: Comprehensive Performance Comparison of Three Experiments Across Different Order Scales

		REWARD		Courier Cost		Satisfaction		Performance Gain
Order Scale	Experiment & Config	Avg.	Rel. Imp.	Avg.	Change	Avg.	Improve.	vs Base
200 Orders								
Exp 1	Random	−58.1	–	26.6	–	85.24%	–	–
	OPC baseline	−64.7	-11.4%	17.8	-33.1%	78.55%	-6.69pp	-11.36%
	Heuristic	−61.7	−6.2%	17.5	−34.2%	83.77%	−1.47*pp*	−1.72%
	Basic DQN	−40.5	+30.3%	28.2	+6.0%	95.12%	+9.88pp	+11.59%
	Basic IQN	−38.9	+33.0%	29.5	+10.9%	96.24%	+11.00pp	+12.90%
Exp 2	MoE-2 Experts	−39.4	–	28.2	–	95.52%	–	–
	MoE-4 Experts	−40.0	−1.5%	28.4	+0.7%	95.36%	−0.16*pp*	−0.17%
	MoE-6 Experts	−38.3	+2.8%	28.5	+1.1%	96.08%	+0.56pp	+0.59%
Exp 3	Extend 30 min	−39.2	–	27.5	–	98.24%	–	–
	Extend 60 min	−37.4	+4.6%	28.1	+2.2%	96.40%	−1.84*pp*	−1.87%
	Extend 90 min	−37.4	+4.6%	27.8	+1.1%	96.16%	−2.08*pp*	−2.12%
	DQN with Extend 60 min	−40.7	-3.3%	28.0	+1.8%	95.12%	−3.12*pp*	−3.18%
500 Orders								
Exp 1	Random	−246.1	–	26.7	–	61.36%	–	–
	OPC baseline	−234.1	+4.9%	28.9	+8.2%	63.47%	+2.57pp	+3.44%
	Heuristic	−210.7	+14.4%	30.8	+15.4%	67.71%	+6.35pp	+10.35%
	Basic DQN	−188.7	+23.3%	39.1	+46.4%	72.08%	+10.72pp	+17.47%
	Basic IQN	−184.5	+25.0%	39.6	+48.3%	73.00%	+11.64pp	+18.97%
Exp 2	MoE-2 Experts	−194.1	–	37.0	–	70.88%	–	–
	MoE-4 Experts	−178.4	+8.1%	39.4	+6.5%	74.20%	+3.32pp	+4.68%
	MoE-6 Experts	−183.1	+5.7%	38.3	+3.5%	73.10%	+2.22pp	+3.13%
Exp 3	Extend 30 min	−151.9	–	53.0	–	82.20%	–	–
	Extend 60 min	−125.9	+17.1%	54.2	+2.3%	86.40%	+4.20pp	+5.11%
	Extend 90 min	−120.1	+20.9%	52.7	−0.6%	88.50%	+6.30pp	+7.66%
	DQN with Extend 60 min	−127.0	+16.4%	54.5	+2.8%	86.06%	+3.86*pp*	+4.70%
1000 Orders								
Exp 1	Random	−741.0	–	26.7	–	41.82%	–	–
	OPC baseline	−598.6	+19.2%	34.5	+29.2%	50.87%	+9.05pp	+19.22%
	Heuristic	−654.5	+11.7%	36.7	+37.5%	47.90%	+6.08pp	+14.54%
	Basic DQN	−573.4	+22.6%	35.9	+34.4%	53.00%	+11.18pp	+26.73%
	Basic IQN	−571.2	+22.9%	36.2	+35.6%	53.20%	+11.38pp	+27.21%
Exp 2	MoE-2 Experts	−554.2	–	36.8	–	54.64%	–	–
	MoE-4 Experts	−545.5	+1.6%	38.9	+5.7%	55.44%	+0.80pp	+1.46%
	MoE-6 Experts	−537.7	+3.0%	38.3	+4.1%	56.08%	+1.44pp	+2.64%
Exp 3	Extend 30 min	−479.0	–	63.1	–	62.72%	–	–
	Extend 60 min	−364.1	+24.0%	72.2	+14.4%	71.52%	+8.80pp	+14.03%
	Extend 90 min	−313.3	+34.6%	78.4	+24.2%	78.50%	+15.78pp	+25.16%
	DQN with Extend 60 min	−376.6	+21.4%	70.5	+11.7%	70.14%	+7.42*pp*	+11.83%

Efficiency Regimes and Resource Investment.

High Efficiency (200 Orders, $\eta \approx 1.01$): In under-resourced regimes, a marginal cost increase (+10.9%) yields substantial satisfaction gains (+11.00pp). Here, conservative spending is suboptimal; investment yields unambiguous returns.

Moderate Efficiency (500 Orders, $\eta \approx 0.24$): As complexity grows, returns diminish but remain positive. Random's low expenditure (26.7) results in poor service quality (61.36% satisfaction), whereas IQN's higher investment (+48.3%) effectively purchases capacity, achieving 73.00% satisfaction. The satisfaction gap of +11.64pp confirms that strategic resource allocation prevents service degradation.

Resource Crisis (1000 Orders, $\eta \approx 0.32$): Random exhibits pathological under-provisioning (cost at 26.7), leading to severe service deterioration (41.82% satisfaction). IQN's increased costs (+35.6%) represent a necessary emergency response, with the satisfaction improvement (+11.38pp to 53.20%) confirming that Random's risk-neutral policy fails to capture the high variance of tail risks under extreme demand conditions.

DQN's Cost-Efficiency Profile.

DQN occupies an interesting middle ground between Random's under-investment and IQN's aggressive spending, revealing the cost implications of distributional modeling:

200 Orders ($\eta_{DQN} \approx 1.65$): DQN's marginal cost increase (+6.0%) yields substantial satisfaction gains (+9.88pp), demonstrating high efficiency. This outperforms IQN's ratio ($\eta_{\text{IQN}} \approx 1.01$), suggesting that at small scales, DQN's more conservative policy avoids unnecessary spending while achieving near-optimal results. The higher efficiency indicates that DQN identifies the minimum viable capacity threshold without the "over-cautious" investment that distributional risk-aversion induces.

500 Orders ($\eta_{DQN} \approx 0.23$): DQN's cost increase (+46.4%) closely tracks IQN's (+48.3%), but with slightly lower satisfaction improvement (+10.72pp vs +11.64pp). The similar investment levels suggest both methods correctly identify the need for aggressive capacity augmentation, with IQN's distributional modeling providing a marginal edge in resource allocation precision ($\eta_{\text{IQN}} = 0.24$ vs $\eta_{\text{DQN}} = 0.23$). This near-parity confirms that at medium scales, the primary driver of performance is recognizing the need for investment rather than fine-tuning its magnitude.

1000 Orders ($\eta_{DQN} \approx 0.32$): DQN's efficiency ($\eta = 0.32$) matches IQN's ($\eta = 0.32$), confirming that both methods adopt nearly identical emergency response strategies under system overload. DQN's marginally lower cost (+34.5% vs +35.6%) with comparable satisfaction (+11.18pp vs +11.38pp) suggests that expectation-based Q-learning suffices for clear-cut capacity shortage scenarios, where the optimal action ("hire aggressively") is unambiguous and does not benefit from tail-risk quantification.

Strategic Non-Monotonicity.

The relationship between cost and reward is regime-dependent:

1. *Scarcity (Positive Correlation):* Under-provisioning (e.g., Random) yields high returns on investment when addressed properly.

2. *Surplus (Negative Correlation):* Over-provisioning (e.g., MoE-2 to MoE-4 at 200 orders) incurs costs without service gains (+0.7% cost for −0.16pp satisfaction).

3. *Strategic Dominance (Extend-Shift):* The Extend-Shift action achieves "super-efficiency" ($\eta \to \infty$) at 500 orders by simultaneously reducing costs (-0.6%) and increasing satisfaction (+6.30pp from 30min to 90min). This validates the superiority of zero-latency, targeted interventions over crude hiring.

Conclusion: Courier cost is a strategic resource investment. While expectation-based methods (Random) systematically under-invest, IQN-MoE's distributional perspective justifies higher costs to mitigate high-variance tail risks, adaptively navigating between scarcity and surplus.

5.7 Experimental Conclusions

Through three sets of comparative experiments, we draw the following conclusions.

First, the IQN algorithm, by estimating complete action-value distributions, outperforms Random, Heuristic, and DQN on delivery scheduling tasks with 22.9%–33.0% reward improvement across scales, more stable training, and faster convergence (reaching stability at epoch 12, 20% faster than DQN).

Second, the 4-expert MoE configuration achieves the best balance between performance, stability, and computational efficiency, with experts automatically learning to handle different delivery scenarios. At 500 orders, MoE-4 outperforms MoE-2 by +8.1% in reward and +3.32pp in satisfaction, while reducing computational overhead by 33% compared to MoE-6.

Third, the Extend-Shift action significantly improves system flexibility for peak hours, yielding satisfaction improvements of 2.0–9.5 percentage points over the base IQN configuration across different scales. At 1000 orders with 90-minute extensions, the mechanism achieves 78.50% satisfaction compared to 53.20% without it—a transformative +25.30pp improvement. The agent intelligently leverages this action during peak periods, and the mechanism demonstrates "super-efficiency" at 500 orders by simultaneously reducing costs (-0.6%) while increasing satisfaction (+6.30pp).

Overall, the IQN-MoE model combined with the Extend-Shift action provides an efficient, stable, and interpretable solution for courier scheduling in food delivery scenarios. The framework achieves up to 33.0% reward improvement over baselines and maintains high satisfaction rates (up to 98.24% at small scales), validating the effectiveness of distributional reinforcement learning for risk-sensitive resource allocation.

6 Conclusion and Future Work

We presented IQN-MoE, a framework synergizing distributional reinforcement learning with regime-adaptive experts to resolve the conflict between stochastic demand and rigid resource allocation in courier scheduling.

Core Contributions & Findings. First, we established that distributional value estimation is superior to rule-based methods (Random, Heuristic) and expectation-based RL (DQN) in high-variance logistics, achieving substantially improved training stability and 20% faster convergence (epoch 12 vs. epoch 15 for DQN). Second, we introduced the **Extend-Shift**

mechanism, demonstrating that domain-aware action space augmentation—specifically, zero-latency capacity correction—provides transformative performance gains during demand surges, achieving up to 34.6% reward improvement and 25.30 percentage point satisfaction increase at large scales (1000 orders). Third, ablation studies identified the four-expert configuration as the optimal Pareto frontier between regime specialization and computational efficiency. Empirically, the framework yields double-digit gains in cumulative rewards (up to 33.0% improvement) while maintaining high order completion rates (up to 98.24%), validating the necessity of the proposed risk-sensitive control paradigm.

Limitations & Future Directions. Several avenues remain for future research. (1) *Sim-to-Real Transfer*: Bridging the gap between simulation and physical deployment requires robust domain adaptation to handle real-world complexities such as traffic variability, courier availability uncertainty, and operational constraints not captured in our current model. (2) *Dynamic Architecture*: Replacing fixed experts with adaptive architectures (e.g., neural architecture search conditioned on workload complexity) could enable automatic capacity scaling based on scenario difficulty. (3) *Spatial Scalability*: Extending the framework to city-scale optimization via hierarchical or federated learning is essential for multi-region coordination, where different zones may operate under distinct demand patterns and resource constraints. (4) *Proactive Integration*: Incorporating explicit demand forecasting (e.g., probabilistic time-series models) into the MoE gating mechanism could enable anticipatory expert routing before regime shifts occur, further shifting the paradigm from reactive to predictive scheduling.

Broader Impact. Beyond food delivery, this work provides a generalizable template for high-stochasticity resource allocation problems. The synergy of distributional RL and mixture-of-experts applies naturally to domains where operational contexts vary significantly and tail risks are non-negligible, including:

Emergency response systems: Ambulance and fire truck dispatch under uncertain incident locations and severity levels;
Cloud computing: Virtual machine allocation and autoscaling under bursty workloads with strict SLA requirements;
Healthcare staffing: Nurse and physician scheduling with stochastic patient arrivals and acuity variations;
Ride-hailing platforms: Dynamic driver positioning and surge pricing under fluctuating rider demand.

The integration of risk-aware distributional modeling with context-adaptive experts offers a robust infrastructure for sequential decision-making in complex, uncertain environments.generation delivery platform optimization and broader stochastic resource allocation problems.

References

[1] Ulmer, M. W., Thomas, B. W., Campbell, A. M., & Woyak, N. (2021). The restaurant meal delivery problem: Dynamic pick-up and delivery with deadlines and random ready times. *Transportation Science*, 55(1), 75–100.

[2] Reyes, D., Erera, A. L., Savelsbergh, M., Sahasrabudhe, S., & O'Neil, R. (2018). The meal delivery routing problem. *Optimization Online*.

[3] Yildiz, B., & Savelsbergh, M. (2019). Provably high-quality solutions for the meal delivery routing problem. *Transportation Science*, 53(5), 1372–1388.

[4] Steever, Z., Karwan, M., & Murray, C. (2019). Dynamic courier routing for a food delivery service. *Computers & Operations Research*, 107, 173–188.

[5] Chen, C., Demir, E., Huang, Y., & Qiu, R. (2021). The adoption of self-driving delivery robots in last mile logistics: A systematic review. *Transportation Research Part E: Logistics and Transportation Review*, 146, 102214.

[6] Bandi, C., & Gupta, D. (2020). Operating room staffing and scheduling. *Manufacturing & Service Operations Management*, 22(5), 958–974.

[7] Dönmez, S., Koç, Ç., & Altıparmak, F. (2022). The mixed fleet vehicle routing problem with partial recharging by multiple chargers: Mathematical model and adaptive large neighborhood search. *Transportation Research Part E: Logistics and Transportation Review*, 167, 102917.

[8] Hu, M., Tan, G.-Z., & Wu, Z.-C. (2025). Adaptive hybrid optimization for integrated project scheduling and staffing problem with time/resource trade-offs. *Operations Research Perspectives*, 15, 100305.

[9] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529–533.

[10] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.

[11] Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning* (pp. 1861–1870). PMLR.

[12] Fujimoto, S., Hoof, H., & Meger, D. (2018). Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning* (pp. 1587–1596). PMLR.

[13] Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., & Dormann, N. (2021). Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268), 1–8.

[14] Stooke, A., Mahajan, A., Barros, C., Deck, C., Bauer, J., Sygnowski, J., Trebacz, M., Jaderberg, M., Mathieu, M., McAleese, N., Bradley-Schmieg, N., Wong, N., Porcel, N.,

Raileanu, R., Hughes-Fitt, S., Dalibard, V., & Fergus, R. (2021). Decoupling representation learning from reinforcement learning. In *International Conference on Machine Learning* (pp. 9870–9879). PMLR.

[15] Dabney, W., Ostrovski, G., Silver, D., & Munos, R. (2018). Implicit quantile networks for distributional reinforcement learning. In *International Conference on Machine Learning* (pp. 1096–1105). PMLR.

[16] Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., & Dean, J. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. arXiv preprint arXiv:1701.06538.

[17] Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation*, 3(1), 79–87.

[18] Ulmer, M. W., & Savelsbergh, M. (2020). Workforce scheduling in the era of crowdsourced delivery. *Transportation Science*, 54(4), 1113–1133.

[19] Luy, A., Ulmer, M. W., & Thomas, B. W. (2024). Strategic workforce planning for crowdsourced delivery platforms with hybrid driver fleets. *Transportation Research Part E: Logistics and Transportation Review*, 181, 103355.

[20] Behrendt, T., Savelsbergh, M., & Wang, X. (2022). Sample average approximation for stochastic empty container repositioning. *Computers & Operations Research*, 140, 105644.

[21] Auad, M., Savelsbergh, M., & Wang, X. (2024). Deep reinforcement learning for dynamic courier hiring in last-mile delivery. *European Journal of Operational Research*, 312(1), 245–259.

[22] Saleh, M., Ulmer, M. W., & Thomas, B. W. (2024). Enhancing last-mile delivery: An agent-based reinforcement learning approach for dynamic shift extensions. *Transportation Research Part C: Emerging Technologies*, 162, 104585.

[23] Saleh, M., Ulmer, M. W., & Thomas, B. W. (2025). Comparing reinforcement learning algorithms for real-time courier scheduling in on-demand delivery. *Computers & Operations Research*, 165, 106571.

[24] Beliën, J., Demeulemeester, E., De Bruecker, P., Van den Bergh, J., & Cardoen, B. (2012). Improving workforce scheduling of aircraft line maintenance at Sabena Technics. *Interfaces*, 42(4), 352–364.

[25] Alfares, H. K. (2004). Survey, categorization, and comparison of recent tour scheduling literature. *Annals of Operations Research*, 127(1), 145–175.

[26] Ernst, A. T., Jiang, H., Krishnamoorthy, M., & Sier, D. (2004). Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, 153(1), 3–27.

[27] Atlason, J., Epelman, M. A., & Henderson, S. G. (2008). A unified approach to simulation-optimization with application to staffing. *Operations Research*, 56(1), 145–160.

[28] Gendreau, M., Hertz, A., & Laporte, G. (1994). A tabu search heuristic for the vehicle routing problem with time windows. *Transportation Science*, 28(3), 185–200.

[29] Ropke, S., & Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4), 455–472.

[30] Cordeau, J.-F., Laporte, G., & Mercier, A. (2001). A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational Research Society*, 52(8), 928–936.

[31] Vidal, T., Crainic, T. G., Gendreau, M., Lahrichi, N., & Rei, W. (2013). A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Computers & Operations Research*, 40(1), 475–489.

[32] Van Hasselt, H., Guez, A., & Silver, D. (2016). Deep reinforcement learning with double Q-learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1).

[33] Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., & Silver, D. (2018). Rainbow: Combining improvements in deep reinforcement learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

[34] Bellemare, M. G., Dabney, W., & Munos, R. (2017). A distributional perspective on reinforcement learning. In *International Conference on Machine Learning* (pp. 449–458). PMLR.

[35] Dabney, W., Rowland, M., Bellemare, M. G., & Munos, R. (2018). Distributional reinforcement learning with quantile regression. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

[36] Dabney, W., Ostrovski, G., Silver, D., & Munos, R. (2018). Implicit quantile networks for distributional reinforcement learning. In *International Conference on Machine Learning* (pp. 1096–1105). PMLR.

[37] Mao, H., Schwarzkopf, M., Venkatakrishnan, S. B., Meng, Z., & Alizadeh, M. (2019). Learning scheduling algorithms for data processing clusters. In *Proceedings of the ACM Special Interest Group on Data Communication* (pp. 270–288).

[38] Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation*, 3(1), 79–87.

[39] Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., & Dean, J. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. arXiv preprint arXiv:1701.06538.

[40] Devin, C., Gupta, A., Darrell, T., Abbeel, P., & Levine, S. (2017). Learning modular neural network policies for multi-task and multi-robot transfer. In *IEEE International Conference on Robotics and Automation* (pp. 2169–2176). IEEE.

[41] Yang, R., Xu, X., Gao, Y., Zhang, R., Ma, H., & Qiao, Y. (2020). Multi-task reinforcement learning with soft modularization. In *Advances in Neural Information Processing Systems*, 33, 4767–4777.

[42] Auad, R., Erera, A., & Savelsbergh, M. (2023). Dynamic courier capacity acquisition in rapid delivery systems: A deep Q-learning approach. *Transportation Science*, 58(1), 67–93.