Deep Reinforcement Learning for Resource Management in IoT-Edge-Cloud Computing Continuum: A Taxonomy and Future Directions

ZHIYU WANG*, The University of Melbourne, Australia
MOHAMMAD GOUDARZI, Monash University, Australia
MINGMING GONG, The University of Melbourne, Australia
RAJKUMAR BUYYA, The University of Melbourne, Australia

The Internet of Things (IoT)-Edge-Cloud computing continuum demands intelligent resource management to satisfy stringent latency, energy, and quality-of-service requirements. Deep Reinforcement Learning (DRL) has emerged as a promising paradigm for adaptive resource management, capable of learning effective policies without requiring accurate analytical models. However, existing surveys lack a unified framework that systematically organizes approaches across fundamental design dimensions. This survey proposes a novel two-dimensional taxonomy that orthogonally separates *control scope* (single-agent versus multi-agent) from *training paradigm* (standard versus federated), resolving ambiguities in prior work. We provide a comprehensive literature review, comparative analysis, and practical design guidelines mapping deployment requirements to algorithmic choices. We further survey advanced enhancement techniques and identify critical open challenges with future research directions toward intelligent, scalable, and privacy-preserving resource management.

CCS Concepts: • **General and reference** → **Surveys and overviews**; • **Computer systems organization** → *Distributed architectures*; • **Computing methodologies** → *Reinforcement learning*.

Additional Key Words and Phrases: Resource Management, Internet of Things, Edge Computing, Cloud computing, Deep Reinforcement Learning, Federated Learning

1 Introduction

The integration of the Internet of Things (IoT) with edge and cloud computing has created a paradigm shift in distributed computing architectures. With IoT deployments reaching unprecedented scale worldwide and generating massive data volumes, traditional cloud-centric architectures face fundamental limitations in serving latency-sensitive applications [173]. Emerging use cases such as autonomous vehicles, augmented reality, and industrial control systems require near-instantaneous response times that cannot tolerate the round-trip delays inherent to distant cloud data centers [154].

The IoT-Edge-Cloud continuum has emerged as the architectural response, forming a three-tier distributed computing hierarchy that bridges resource-constrained IoT devices with resource-abundant cloud datacenters through intermediate edge infrastructure deployed at network edges [59]. By distributing computation across this hierarchy, the continuum enables latency-critical processing at the edge while leveraging cloud resources for compute-intensive analytics, creating a distributed computing fabric spanning billions of heterogeneous devices across global geographical scales [173].

Efficient resource management across this continuum is both critical and challenging. System operators must dynamically allocate computational resources, schedule heterogeneous workloads, and make intelligent task placement decisions under stringent constraints [175]. The challenges are multifaceted: unprecedented scale, extreme dynamics, pervasive heterogeneity spanning device capabilities and network technologies, and strict operational constraints including latency bounds and energy budgets [154]. Moreover, privacy regulations and data sovereignty requirements

Authors' Contact Information: Zhiyu Wang, zhiywang1@student.unimelb.edu.au, The University of Melbourne, Melbourne, Victoria, Australia; Mohammad Goudarzi, mohammad.goudarzi@monash.edu, Monash University, Melbourne, Victoria, Australia; Mingming Gong, mingming.gong@unimelb.edu.au, The University of Melbourne, Melbourne, Victoria, Australia; Rajkumar Buyya, rbuyya@unimelb.edu.au, The University of Melbourne, Melbourne, Victoria, Australia.

increasingly mandate that sensitive data remains local to edge nodes, prohibiting centralized data collection for training [179]. Inefficient resource allocation directly translates to quality-of-service violations, excessive energy consumption, and substantial operational costs for large-scale deployments [175, 201].

Deep Reinforcement Learning (DRL) has emerged as a promising paradigm for adaptive resource management in this complex environment. Unlike traditional rule-based heuristics relying on manually designed policies or model-based optimization requiring accurate analytical models, DRL agents learn near-optimal policies through environmental interaction without prior system models [116]. This model-free adaptive learning capability makes DRL particularly suited to the IoT-Edge-Cloud continuum, where workload patterns are unpredictable, system dynamics are non-stationary, and accurate models are difficult to obtain [15]. However, standard DRL training assumes unrestricted data access, which conflicts with privacy regulations and data sovereignty requirements in IoT/edge environments [162]. Federated learning addresses this challenge by enabling collaborative model training across distributed nodes/regions while keeping data local, making it an essential training paradigm for privacy-preserving resource management [216]. Research in this area has grown rapidly, motivating the need for a comprehensive survey that systematically organizes the expanding literature.

1.1 Related Surveys and Research Gap

Resource management in the IoT-Edge-Cloud continuum has generated numerous surveys, but these works exhibit significant limitations in scope definition, methodological organization, and analytical depth. Table 1 systematically compares 16 representative surveys published between 2023 and 2025, evaluating them across two dimensions: **method coverage** (breadth of methods and architectures discussed) and **analytical components** (depth of system-level and methodology-level analysis provided), revealing substantial gaps in existing literature.

Single-Agent Dominance with Weak Multi-Agent Coverage. Among the 16 surveys, 12 fully cover serial Single-Agent Reinforcement Learning (SARL) methods (e.g., Deep Q-Network (DQN)), but only 4 partially address parallel SARL (e.g., Asynchronous Advantage Actor-Critic (A3C)). Multi-agent coverage is extremely weak: only 2 surveys mention independent Multi-agent Reinforcement Learning (MARL) (e.g., Independent DQN (IDQN)), and only Yang et al. [189] covers Centralized Training with Decentralized Execution (CTDE) methods (e.g., Multi-Agent Deep Deterministic Policy Gradient (MADDPG)). This gap is critical given that the IoT-Edge-Cloud continuum is inherently a distributed system where multi-agent architectures naturally map to autonomous edge nodes.

Nearly Complete Absence of Federated Training Paradigm. Only 3 surveys provide comprehensive coverage of centralized federated learning, with 6 offering partial treatment. Most critically, no survey covers decentralized federated architectures, despite their crucial trade-offs in convergence speed, scalability, and fault tolerance for large-scale deployments. Furthermore, existing surveys treat federated learning as an isolated technique rather than a training paradigm orthogonally combinable with different control architectures, leaving federated single-agent and federated multi-agent design spaces unexplored.

Lacking System-Level Analysis across Key Dimensions. Existing surveys provide insufficient system-level analysis across scalability, convergence, overhead, and adaptability. Only Yang et al. [189] offers a comprehensive analysis of scalability, with the majority providing at most partial coverage. No survey comprehensively examines convergence properties or overhead trade-offs, despite these being critical differentiators between standard and federated training paradigms as well as between centralized and decentralized architectures. This gap leaves practitioners without

Table 1. Comparison of existing surveys on resource management in IoT/edge/cloud computing environments (Ordered by Year).

Survey	**Year**	**Method Coverage**						**Analytical Components**					
		Standard				**Federated**		**System-Level**				**Methodology-Level**	
		SARL		**MARL**		**Centr-alized**	**Decentr-alized**	**Scalability**	**Convergence**	**Overhead**	**Adaptability**	**DRL Enhancement Techniques**	**Deployment Guidelines**
		Serial	**Parallel**	**Independent**	**CTDE**								
Zhang et al. [212]	2023	✓	×	×	×	✓	×	×	∘	∘	∘	×	∘
Chiang et al. [23]	2023	✓	∘	×	×	∘	×	×	×	∘	∘	×	∘
Wang et al. [172]	2023	✓	×	×	×	×	×	×	×	×	∘	×	∘
Hortelano et al. [64]	2023	✓	∘	×	×	∘	×	×	×	×	∘	×	×
Zabihi et al. [203]	2024	✓	∘	×	×	∘	×	∘	∘	×	∘	∘	×
Walia et al. [167]	2024	✓	×	×	×	✓	×	∘	×	∘	∘	∘	✓
Tang et al. [158]	2024	✓	×	∘	×	∘	×	∘	∘	∘	∘	✓	∘
Dong et al. [32]	2024	✓	×	×	×	×	×	×	×	×	×	×	×
Asghari et al. [7]	2024	∘	×	×	×	×	×	×	×	×	×	×	×
Maia et al. [113]	2024	∘	×	×	×	×	×	×	×	×	×	×	×
Zolghadri et al. [222]	2024	∘	×	×	×	×	×	×	×	×	×	×	×
Taleb et al. [154]	2025	✓	×	×	×	×	×	×	×	×	∘	×	∘
Rasouli et al. [138]	2025	∘	×	×	×	∘	×	×	×	×	×	×	×
Vetriveeran et al. [166]	2025	✓	×	×	×	×	×	∘	×	×	∘	×	×
Yang et al. [189]	2025	✓	∘	∘	∘	∘	×	✓	×	×	×	∘	×
Wu et al. [183]	2025	✓	×	×	×	✓	×	∘	×	∘	∘	✓	×
This survey	**2026**	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Note: ✓: Fully covered; ∘: Partially covered; ×: Not covered. **SARL**: Single-Agent RL; **MARL**: Multi-Agent RL; **CTDE**: Centralized Training with Decentralized Execution.

actionable guidance on how different DRL configurations behave under the dynamic workloads, heterogeneous resources, and communication constraints characteristic of the IoT-Edge-Cloud continuum.

Inadequate Methodology-Level Analysis. At the methodology level, only 5 surveys cover DRL enhancement techniques (e.g., specialized network architectures, training optimizations, model compression, and safety mechanisms) that are essential for practical deployment on resource-constrained edge devices. More critically, only Walia et al. [167] provides comprehensive deployment guidelines, with 5 offering partial guidance. No survey provides a systematic mapping from deployment requirements (scale, privacy constraints, resource budgets) to appropriate DRL configurations across the two-dimensional space of control architectures and training paradigms, making it difficult for system designers to select suitable methods for their specific scenarios.

As shown in Table 1, to the best of our knowledge, no prior survey simultaneously provides comprehensive coverage across both dimensions of control scope (serial/parallel SARL, independent/CTDE MARL) and training paradigms (standard, centralized/decentralized federated), introduces a principled orthogonal taxonomy, provides systematic system-level analysis across scalability, convergence, overhead, and adaptability, offers practical deployment guidelines mapping requirements to algorithmic choices, comprehensively reviews enhancement techniques, and provides in-depth analysis of emerging paradigms. By systematically addressing the fragmentation in existing literature, this survey establishes a comprehensive foundation for DRL-based resource management in the IoT-Edge-Cloud continuum.

1.2 Survey Contributions and Scope

This survey makes the following contributions:

(1) Two-Dimensional Taxonomy. We propose a two-dimensional taxonomy that orthogonally separates control scope (single-agent vs. multi-agent) from training paradigm (standard vs. federated), resolving ambiguities in prior surveys that conflate these independent design dimensions.

(2) Comprehensive Literature Review. We systematically review over 100 papers applying DRL to resource management in the IoT-Edge-Cloud computing continuum (2020-early 2026), organizing them through our two-dimensional taxonomy.

(3) System-Level Analysis and Design Guidelines. We systematically analyze and compare methods across scalability, convergence, overhead, and adaptability, and provide practical design guidelines mapping deployment requirements to appropriate algorithmic choices.

(4) Advanced Techniques Survey. We survey orthogonal enhancement techniques including specialized network architectures, training optimizations, model compression, and safety mechanisms that apply across taxonomy categories.

(5) Open Challenges and Future Directions. We identify critical research challenges where substantial gaps remain and outline promising directions for future research.

Scope. This survey covers DRL-based methods for resource management in the IoT-Edge-Cloud continuum (2020-early 2026). We exclude non-DRL-based methods and pure networking problems without explicit resource management objectives.

1.3 Organization

The rest of this survey is organized as follows. Section 2 provides background on the IoT-Edge-Cloud continuum, DRL fundamentals, and problem formulation. Section 3 presents our two-dimensional taxonomy. Sections 4 and 5 systematically review DRL approaches under standard and federated training paradigms, each organized by control scope (SARL then MARL). Section 6 surveys advanced techniques. Section 7 discusses open challenges and future directions. Section 8 provides the conclusion.

2 Background and Preliminaries

This section establishes the foundational concepts for DRL-based resource management in the IoT-Edge-Cloud continuum, covering the computing architecture, DRL fundamentals, and problem formulation.

2.1 IoT-Edge-Cloud Continuum

The IoT-Edge-Cloud continuum is a hierarchical computing paradigm that bridges resource-constrained IoT devices with cloud datacenters through intermediate edge infrastructure [59, 154]. Unlike cloud-centric architectures, this continuum distributes workloads across three tiers based on latency requirements, resource availability, and data locality [175].

As illustrated in Figure 1, the *cloud layer* provides virtually unlimited computational resources for intensive analytics and model training [154]. The *edge layer*, comprising base stations, cloudlets, and micro-datacenters, offers moderate resources with low-latency access to end devices [154]. The *IoT layer* encompasses heterogeneous devices with constrained computational capacity, memory, energy budgets, and intermittent connectivity [5, 81]. Many IoT applications exhibit Directed Acyclic Graph (DAG)-structured task dependencies requiring precedence-aware scheduling [162, 197].

This architecture presents three fundamental challenges that motivate learning-based approaches:

Dynamic workloads. IoT applications generate highly variable traffic patterns with temporal dynamics and spatial heterogeneity [175]. User mobility further exacerbates unpredictability [92], while DAG-structured dependencies impose time-varying execution constraints [162]. Static policies cannot adapt to such variability.

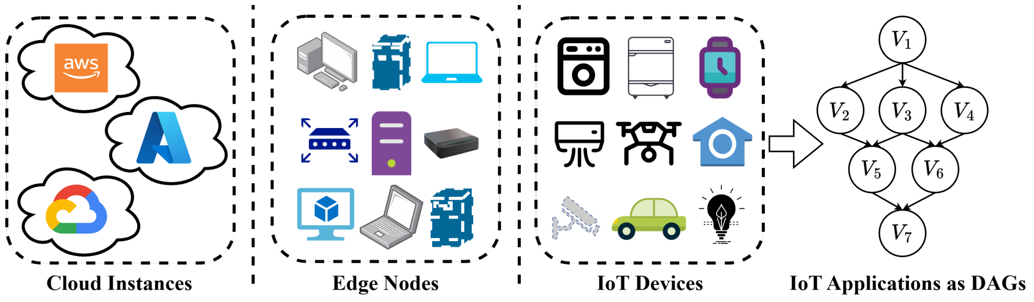

Fig. 1. The IoT-Edge-Cloud continuum architecture.

Pervasive heterogeneity. The continuum spans devices with vastly different capabilities, network technologies with variable bandwidth and latency, and applications with fundamentally different Quality of Service (QoS) requirements [149, 162, 175]. Manual policy design for each scenario is infeasible [176].

Model uncertainty. Stochastic task arrivals, network fluctuations, and device failures render accurate system modeling intractable [154, 162]. Optimization approaches assuming known models suffer severe performance degradation under such uncertainty.

These challenges, namely dynamism, heterogeneity, and uncertainty, necessitate adaptive approaches capable of learning effective policies through environmental interaction, motivating the DRL methods formalized next.

2.2 Deep Reinforcement Learning Fundamentals

DRL provides a mathematical framework for sequential decision-making under uncertainty, where an agent learns optimal behavior through trial-and-error interaction with an environment [153]. Unlike supervised learning that requires labeled examples of correct decisions, or unsupervised learning that discovers patterns in unlabeled data, DRL agents learn from evaluative feedback signals indicating action quality, making it particularly suited to resource management scenarios where optimal policies are unknown *a priori* but system performance can be measured.

The reinforcement learning problem is formalized as a **Markov Decision Process (MDP)**, defined by the tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$:

- $\mathcal{S}$ is the **state space**, the set of all possible environmental configurations that fully describe the system at any time instant.
- $\mathcal{A}$ is the **action space**, the set of decisions available to the agent.
- $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the **state transition probability function**, where $P(s'|s, a)$ specifies the probability of transitioning to state s' after executing action a in state s.
- $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the **reward function**, providing scalar feedback $r = R(s, a)$ that evaluates action quality.
- $\gamma \in [0, 1]$ is the **discount factor**, determining the relative importance of immediate versus future rewards.

A **policy** $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ maps states to probability distributions over actions, where $\Delta(\mathcal{A})$ denotes the probability simplex over the action space. Deterministic policies $\pi : \mathcal{S} \to \mathcal{A}$ select actions without randomization. The agent's objective is to discover a policy that maximizes the expected cumulative discounted reward, formally expressed as:

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

where $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \ldots)$ denotes a trajectory sampled by executing policy π.

Two fundamental constructs enable policy optimization:

The **state value function** $V^\pi(s)$ represents the expected cumulative reward starting from state s and following policy π:

$$V^\pi(s) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r_t \mid s_0 = s\right]. \tag{2}$$

The **state-action value function** (Q-function) $Q^\pi(s,a)$ extends this to state-action pairs:

$$Q^\pi(s,a) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r_t \mid s_0 = s, a_0 = a\right]. \tag{3}$$

The optimal Q-function $Q^*(s,a) = \max_\pi Q^\pi(s,a)$ satisfies the **Bellman optimality equation**:

$$Q^*(s,a) = \mathbb{E}_{s'\sim P(\cdot|s,a)}\left[R(s,a) + \gamma \max_{a'} Q^*(s',a')\right], \tag{4}$$

and enables optimal policy extraction via $\pi^*(s) = \arg\max_a Q^*(s,a)$.

Policy optimization can be achieved through three paradigmatic approaches. *Value-based methods* learn value functions (typically Q-functions) and derive policies through value maximization. *Policy gradient methods* directly parameterize and optimize policies using gradient ascent on the objective $J(\pi)$. *Actor-critic methods* maintain explicit representations of both policy (actor) and value function (critic), combining the benefits of both paradigms.

DRL employs deep neural networks as function approximators for policies $\pi_\theta(a|s)$, value functions $V_\theta(s)$, or Q-functions $Q_\theta(s,a)$, where θ denotes learnable parameters. Neural networks enable generalization across large or continuous state-action spaces intractable for tabular representations, automatic feature extraction from high-dimensional raw observations, and representation learning that discovers compact encodings of complex state spaces.

In multi-agent settings where multiple decision-makers interact simultaneously, the framework extends to **Decentralized Partially Observable MDPs (Dec-POMDPs)**, defined by the tuple $(\mathcal{N}, \mathcal{S}, \{\mathcal{A}_i\}, \{O_i\}, P, O, \{R_i\}, \gamma)$, where $\mathcal{N} = \{1, \ldots, n\}$ is the set of agents, $\mathcal{S}$ is the global state space, $\mathcal{A}_i$ and O_i are agent i's action and observation spaces, $P(s'|s, \mathbf{a})$ is the joint transition function conditioned on the joint action $\mathbf{a} = (a_1, \ldots, a_n)$, $O(o_i|s', \mathbf{a})$ is the observation function that determines what each agent perceives after a transition, and R_i is agent i's reward function. Each agent i receives local observations $o_i \in O_i$ (which may be partial views of the global state), executes actions $a_i \in \mathcal{A}_i$, and receives individual rewards r_i. The joint action of all agents determines state transitions and rewards, creating complex strategic interactions. Multi-agent learning approaches range from independent learning (where each agent independently employs single-agent RL, treating others as part of the environment) to coordinated learning (leveraging centralized training mechanisms to discover cooperative strategies).

2.3 Resource Management Problem Formulation

Resource management in the IoT-Edge-Cloud continuum can be naturally formulated as MDPs where the system learns to make allocation, scheduling, and offloading decisions to optimize performance metrics.

State space design. The state s_t at time t typically aggregates three categories of information: (i) *System state*: Resource utilization metrics (CPU load, memory occupancy, storage capacity), energy levels (battery charge, power consumption), queue lengths (waiting tasks, buffered data), and device operational status (availability, failure indicators). (ii) *Workload state*: Task characteristics (arrival rates, computational demands, input data sizes, deadline requirements), application types and priorities, and user mobility patterns (in mobile edge scenarios). (iii) *Network state*: Available bandwidth on communication links, end-to-end latencies, wireless channel quality indicators, link reliability metrics, and network topology (in dynamic environments).

Action space design. The action a_t specifies resource management decisions, which may be discrete, continuous, or hybrid depending on the problem: (i) *Discrete actions*: Binary offloading decisions (execute locally or remotely), server selection among N edge nodes, task priority assignments, or cache admission/eviction decisions. (ii) *Continuous actions*: Resource allocation percentages (CPU share $\in [0, 1]$, memory quota), transmission power levels, bandwidth allocation ratios, or voltage/frequency scaling parameters. (iii) *Hybrid actions*: Combinations such as selecting an edge server (discrete) followed by specifying resource allocation quantities (continuous).

Reward function design. The reward r_t encodes optimization objectives through scalar feedback. Common formulations include:

(i) *Single-objective*: Negative latency for delay minimization, negative energy for power efficiency, or cache hit rate for caching problems:

$$r = -\alpha \cdot \text{delay}, \quad r = -\beta \cdot \text{energy}, \quad r = \mathbb{I}[\text{cache hit}], \tag{5}$$

where $\mathbb{I}[\cdot]$ is the indicator function returning 1 if the condition is true and 0 otherwise.

(ii) *Multi-objective*: Weighted combinations with coefficients α, β, ω reflecting relative importance:

$$r = -\alpha \cdot \text{latency} - \beta \cdot \text{energy} + \omega \cdot \text{throughput}. \tag{6}$$

(iii) *Constraint penalties*: Additive penalty terms for violations such as overload prevention or deadline enforcement:

$$r_{\text{overload}} = r_{\text{base}} - \lambda \cdot \max(0, \text{ load} - \text{capacity}), \tag{7}$$

$$r_{\text{deadline}} = r_{\text{base}} - \mu \cdot \mathbb{I}[\text{deadline missed}], \tag{8}$$

where λ and μ are penalty coefficients controlling the severity of constraint violations.

Several challenges complicate the formulation of resource management problems in the IoT-Edge-Cloud continuum as MDPs:

Partial observability. Individual edge nodes typically lack complete visibility into global system state, observing only local metrics and neighboring conditions. This necessitates decentralized multi-agent formulations where agents make decisions based on local observations, or recurrent neural network architectures that infer hidden state from observation histories.

Hybrid action spaces. Many problems require simultaneous discrete decisions (e.g., server selection) and continuous control (e.g., resource allocation percentages). Standard value-based methods handle discrete actions naturally but struggle with continuous spaces, while policy gradient and actor-critic methods accommodate both but introduce training complexity.

Multi-objective optimization. IoT-Edge-Cloud scenarios often involve conflicting objectives: minimizing latency may increase energy consumption, maximizing throughput may degrade fairness. Reward engineering through weighted combinations requires careful tuning, while constrained reinforcement learning formulations provide principled frameworks for handling hard constraints.

Privacy and data sovereignty. Privacy regulations (e.g., GDPR, HIPAA) and data sovereignty requirements mandate that sensitive data remains within specific jurisdictions or local nodes. This prohibits formulation approaches that assume centralized data collection or unrestricted sharing of training experiences across the continuum, requiring federated learning paradigms that enable collaborative policy training while keeping raw data local at each node.

Table 2. Two-dimensional taxonomy framework for DRL-based resource management in IoT-Edge-Cloud continuum.

Control Scope / **Training Paradigm**	**Single-Agent Architecture (SARL)**	**Multi-Agent Architecture (MARL)**
Standard Training (Fig. 2)	§4.1	§4.4
Federated Training (Fig. 3)	§5.1.2, §5.4.2	§5.1.3, §5.4.3

3 Taxonomy and Classification Methodology

We propose a two-dimensional taxonomy that orthogonally separates control scope from training paradigm for DRL-based resource management in the IoT-Edge-Cloud continuum. This framework enables systematic comparison of methods based on:

- **Dimension I: Control Scope**. This dimension distinguishes approaches based on the number of autonomous decision-making entities:
  - *Single-Agent Architecture (SARL)*: A unified global policy is learned and deployed to control resource management across the entire IoT-Edge-Cloud system. While training may involve multiple distributed workers (e.g., A3C) to accelerate convergence, all system components ultimately execute the same shared policy.
  - *Multi-Agent Architecture (MARL)*: Multiple autonomous agents are deployed across the system (e.g., one per edge computing node or region), each learning and executing its own policy. Agents may operate independently (e.g., IDQN) or coordinate through cooperation (e.g., MADDPG).
- **Dimension II: Training Paradigm**. This dimension categorizes approaches based on whether federated learning is employed:
  - *Standard Training*: No privacy constraints on data sharing. Training data, intermediate results (e.g., gradients, experiences), or model parameters can be freely exchanged between computing nodes/regions.
  - *Federated Training*: Training under strict data locality constraints for privacy preservation. Raw training data remains local at each computing node/region. Nodes exchange only model information (e.g., parameters in Federated Averaging (FedAvg)) through federated aggregation mechanisms, preserving data sovereignty. Implementation can be through centralized architectures (with a central server) or decentralized architectures (peer-to-peer coordination).

Table 2 summarizes the resulting taxonomy framework. The two dimensions are orthogonal: any control scope (SARL or MARL) can be combined with any training paradigm (Standard or Federated), yielding distinct architectural patterns with different deployment requirements and performance characteristics. Detailed taxonomies for each paradigm are presented in Figures 2 and 3.

Concretely, the four combinations yield the following architectural patterns:

- *Standard SARL* (§4.1): A single global policy is trained with unrestricted data access, typically on a centralized controller that observes system-wide state. Training may be accelerated through parallel workers (e.g., A3C, IMPALA), but all workers contribute to the same shared policy.
- *Standard MARL* (§4.4): Multiple agents each learn distinct policies tailored to their local domains. Training may leverage global information during a centralized training phase (CTDE), but each agent ultimately executes its own independent policy.

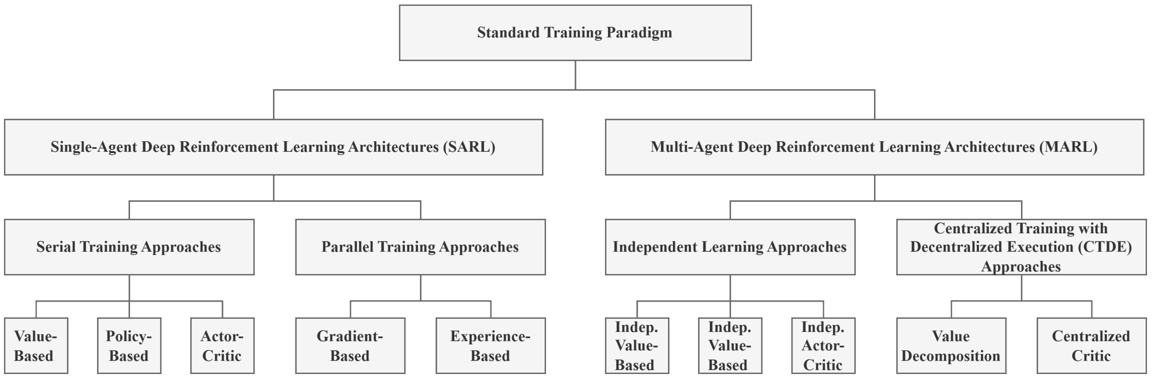
Standard Training Paradigm
Single-Agent Deep Reinforcement Learning Architectures (SARL)
Multi-Agent Deep Reinforcement Learning Architectures (MARL)
Serial Training Approaches
Parallel Training Approaches
Independent Learning Approaches
Centralized Training with Decentralized Execution (CTDE) Approaches
Value-Based
Policy-Based
Actor-Critic
Gradient-Based
Experience-Based
Indep. Value-Based
Indep. Value-Based
Indep. Actor-Critic
Value Decomposition
Centralized Critic

Fig. 2. Taxonomy of DRL methods under standard training paradigm.

- *Federated SARL* (§5.1.2, §5.4.2): A single shared policy is collaboratively trained across distributed environments without exchanging raw data. Each participant trains a local copy on private data and periodically contributes model updates through federated aggregation, producing one global policy deployed system-wide.
- *Federated MARL* (§5.1.3, §5.4.3): Multiple agents learn distinct per-agent policies under data locality constraints. Federated aggregation operates over each agent's policy (and coordination structures in CTDE), enabling knowledge sharing across distributed environments while preserving both agent autonomy and data privacy.

A key distinction emerges among three paradigms that all involve multiple participants training in parallel. *Standard parallel SARL* uses distributed workers to accelerate training of one shared policy with no restrictions on data access, where the parallelism serves *training efficiency*. *Federated SARL* similarly trains one shared policy across distributed environments, but under strict data locality constraints where each participant can only access its own private data, with the parallelism serving *collaborative consensus under privacy constraints*. *Standard MARL* trains separate per-agent policies, each tailored to its local domain, where the parallelism serves *agent autonomy*.

4 Standard Training Paradigm

The standard training paradigm assumes no privacy constraints on data sharing, enabling free exchange of training data, gradients, experiences, or model parameters between computing nodes. Within this paradigm, we categorize DRL approaches by control scope: *single-agent architectures* employ a unified global policy offering coherent optimization at the cost of potential scalability bottlenecks, while *multi-agent architectures* deploy autonomous agents with individual policies, enabling distributed decision-making that aligns with edge computing infrastructures.

Figure 2 illustrates the taxonomy under this paradigm. Section 4.1 explores single-agent architectures with serial and parallel training approaches. Section 4.4 investigates multi-agent architectures, analyzing independent learning and centralized training with decentralized execution.

4.1 Single-Agent Reinforcement Learning Architectures

This centralized decision-making paradigm is particularly prevalent when a central controller has visibility into system-wide state and global optimization objectives outweigh the benefits of distributed autonomy. We organize SARL methods by their training mechanisms. *Serial Training Approaches* (Section 4.1.1) execute learning on a single computational thread, suitable for small to medium-scale deployments. *Parallel Training Approaches* (Section 4.1.2) leverage distributed workers to accelerate learning, essential for large-scale IoT environments generating high-volume data streams. This dichotomy reflects a fundamental trade-off between implementation simplicity and training scalability.

4.1.1 ***Serial Training Approaches.***

Serial training approaches sequentially collect experiences and update policies on a single computational thread. These approaches remain widely adopted in resource management due to mature implementations, theoretical guarantees, and effectiveness in moderately-sized problem spaces. We organize serial methods by their policy representation: *Value-Based Methods* learn state-action value functions to derive policies through value maximization, *Policy-Based Methods* directly parameterize and optimize policies using gradient ascent, and *Actor-Critic Methods* maintain both explicit policy and value function representations to combine their respective advantages.

Value-Based Methods. Value-based methods learn state-action value functions $Q(s, a)$ that estimate the expected cumulative reward of executing action a in state s, with policies derived through value maximization (e.g., $\pi(s) = \arg\max_a Q(s, a)$). This indirect policy representation is particularly suited to discrete action spaces common in resource allocation decisions.

Deep Q-Network (DQN) [119] pioneered the integration of deep neural networks with Q-learning by introducing experience replay (storing transitions (s, a, r, s') in a replay buffer $\mathcal{D}$ and sampling mini-batches to break temporal correlations) and a separate target network to stabilize training. Subsequent innovations addressed DQN's limitations: Double DQN [165] mitigates overestimation bias by decoupling action selection and evaluation; Dueling DQN [177] decomposes $Q(s, a)$ into state value $V(s)$ and advantage $A(s, a)$ streams, improving learning efficiency when many actions yield similar values; Rainbow [62] synthesizes six orthogonal improvements (Double DQN for overestimation mitigation, dueling architecture for value-advantage decomposition, prioritized experience replay, multi-step learning, categorical value distribution, and noisy networks for exploration), achieving superior sample efficiency.

In IoT-Edge-Cloud resource management, DQN variants have been extensively deployed for discrete decision problems. Xiong et al. [185] applied DQN to minimize the long-term weighted sum of task completion time and resource utilization in IoT edge computing, with further works addressing edge service placement [55], delay-aware energy-efficient offloading [4], and joint optimization of energy, delay, and privacy [207]. Liao et al. [97] employed Double DQN to jointly optimize offloading decisions and resource allocation, balancing delay and energy consumption in Multi-access Edge Computing (MEC), with additional applications in air-ground integrated systems [19] and fault-tolerant service function chain placement [170]. Tadele et al. [10] leveraged Dueling DQN to minimize end-to-end Age of Information (AoI) in vehicular fog systems, with extensions to microservice deployment [40]. Zhang et al. [210] deployed Rainbow DQN to reduce waiting delay and system congestion in Industrial IoT (IIoT) data scheduling, alongside applications in AoI-driven caching [202], vehicular task offloading [37], quantum cloud placement [121], and Unmanned Aerial Vehicle (UAV)-enabled MEC [27].

Despite their effectiveness for discrete decisions, value-based methods face inherent limitations. Action spaces grow exponentially with decision dimensionality (e.g., simultaneously allocating CPU, memory, and bandwidth), causing the curse of dimensionality [98, 153]. Furthermore, these methods cannot directly handle continuous action spaces (e.g., precise power levels, bandwidth percentages) without discretization, which introduces approximation errors and scalability issues [98]. These constraints motivate policy-based approaches that directly parameterize policies over continuous spaces.

Policy-Based Methods. Policy-based methods directly parameterize and optimize policies $\pi_\theta(a|s)$ without explicitly learning value functions, enabling natural handling of continuous and high-dimensional action spaces. Modern policy gradient methods incorporate value function baselines to reduce variance while maintaining the core policy optimization objective.

Trust Region Policy Optimization (TRPO) [143] ensures monotonic policy improvement by constraining the Kullback-Leibler (KL) divergence between successive policies, but its second-order optimization requirements impose substantial computational costs. Proximal Policy Optimization (PPO) [144] simplifies this approach through a clipped surrogate objective limiting the ratio between new and old policies, achieving comparable performance with first-order computation at significantly lower overhead.

In resource management scenarios, these methods have proven effective for continuous action spaces. Priyadarshni et al. [131] employed TRPO within a meta reinforcement learning framework for task offloading in MEC, with further TRPO applications in vehicular networks [128] and network slicing [13]. Wang et al. [176] applied PPO to jointly minimize system load and response time in edge and fog computing, with further PPO applications in computation offloading [169], resource allocation [22], cloud scheduling [74], container placement [90], workload allocation [73], and serverless autoscaling [2].

The stability of PPO's policy updates is particularly valuable in production deployments where catastrophic performance degradation must be avoided. However, both PPO and TRPO remain fundamentally on-policy, requiring fresh samples for each update and continuous environment interaction, which limits sample efficiency compared to off-policy methods that can reuse historical data. This motivates actor-critic architectures that combine policy optimization with off-policy learning.

Actor-Critic Methods. Actor-critic methods maintain explicit parameterizations of both policy (actor) $\pi_\theta(a|s)$ and value function (critic) $Q_\phi(s, a)$ or $V_\psi(s)$. The critic evaluates actions to guide policy improvement, while the actor focuses on policy representation. Off-policy actor-critic methods further enable experience replay, dramatically improving sample efficiency. This is critical for real-world deployments where environment interactions are expensive.

Deep Deterministic Policy Gradient (DDPG) [98] adapts DQN's experience replay and target networks to continuous control by learning a deterministic policy $\mu_\theta(s)$ guided by a critic $Q_\phi(s, a)$, but suffers from overestimation bias and hyperparameter sensitivity. Twin Delayed DDPG (TD3) [41] addresses these issues through twin critics (taking the minimum to reduce overestimation), delayed policy updates, and target policy smoothing (adding noise to target actions for regularization). Soft Actor-Critic (SAC) [51] further improves sample efficiency and stability through maximum entropy reinforcement learning that maximizes both expected return and policy entropy, with automatic temperature tuning providing robustness across diverse tasks without extensive hyperparameter search.

These off-policy actor-critic methods have become the choice for continuous control problems in resource management. Ale et al. [3] employed DDPG for task partitioning and offloading with constrained hybrid action spaces in MEC, with further DDPG applications in Industrial IoT [20], MEC offloading [17], 5G scheduling [48], and healthcare MEC [58]. Li et al. [86] applied an improved TD3 for resource allocation in Integrated Sensing and Communication (ISAC)-aided vehicular edge computing, with further TD3 works on UAV offloading [53], Vehicle-to-Everything (V2X) resource allocation [147], service function chain offloading [35], and vehicular computation offloading [192]. Tang et al. [157] utilized SAC for intelligence sharing across distributed edge nodes, with further SAC applications in vehicular offloading [96], 6G MEC [150], IoT lifetime maximization [60], and UAV-assisted offloading [30].

Comparative studies reveal nuanced trade-offs among these methods. Empirical evaluations demonstrate that TD3 typically achieves higher asymptotic performance through its delayed policy updates and target policy smoothing mechanisms [41], while SAC exhibits superior sample efficiency and training stability via maximum entropy reinforcement learning at the cost of increased computational overhead [51]. Both improve upon DDPG in various continuous control benchmarks [41, 51], with the choice between them depending on the relative importance of final performance versus sample efficiency in the specific application.

4.1.2 ***Parallel Training Approaches.***

As IoT deployments scale to millions of devices generating continuous data streams, single-threaded serial training struggles to meet learning efficiency demands. Parallel training architectures address this challenge by distributing the learning process across multiple workers, enabling both faster convergence through increased data throughput and improved exploration through diverse parallel trajectories. We distinguish two paradigms based on what workers share: *Gradient-Based Parallelism* where workers compute and communicate gradients, and *Experience-Based Parallelism* where workers collect experiences for centralized learning.

Gradient-Based Parallelism. Gradient-based parallel methods distribute both environment interaction and gradient computation across multiple workers that share a global parameter server. Each worker independently collects experiences, computes gradients, and contributes to global model updates, effectively parallelizing the entire learning pipeline.

Asynchronous Advantage Actor-Critic (A3C) [118] pioneered this paradigm by executing multiple workers that asynchronously update shared parameters immediately upon computing gradients. This maximizes hardware utilization and provides implicit exploration diversity, with the advantage function $A^{\pi}(s, a) = Q^{\pi}(s, a) - V^{\pi}(s)$ reducing gradient variance to stabilize learning despite asynchronous updates. However, A3C's asynchrony can cause workers to compute gradients based on stale parameters, potentially degrading sample efficiency. Advantage Actor-Critic (A2C), a synchronous variant, addresses this by batching gradients from all workers before updating, ensuring stable gradient aggregation at the cost of reduced parallelism.

In edge computing scenarios, gradient-based parallelism naturally maps to distributed edge infrastructure. Zou et al. [223] proposed A3C-DO for regional resource scheduling that addresses device heterogeneity through dynamic offloading decisions, with further A3C applications in Industrial IoT scheduling [214], dynamic edge-cloud scheduling [164], vehicular edge computing [76], and MEC-enabled blockchain systems [33]. Lu et al. [109] developed A2C-DRL for dynamic scheduling in stochastic edge-cloud environments under workload uncertainties, with further A2C applications in service caching and offloading [156], service provisioning [93], UAV resource scheduling [71], and latency optimization [188].

The key advantage of gradient-based parallelism lies in its simplicity and direct correspondence to distributed edge deployments. However, the on-policy nature of A3C and A2C limits sample reuse [153], and the communication overhead of frequent gradient exchanges can become problematic in bandwidth-constrained edge environments [99]. These limitations motivate experience-based methods that decouple data collection from learning.

Experience-Based Parallelism. Experience-based parallel methods separate the roles of actors (collecting experiences) and learners (training models), enabling massive parallelization of data collection while centralizing computation-intensive learning. Actors execute policies to gather transitions (s, a, r, s') and send them to a shared replay buffer; learners sample from this buffer to train models off-policy. This decoupling allows actors to use slightly outdated policies, dramatically increasing scalability.

Ape-X [63] scales to hundreds of parallel actors by distributing experience collection through a single prioritized replay buffer, where actors transmit only compact experiences rather than high-dimensional gradients, significantly reducing communication overhead. Actors periodically synchronize with the latest policy and contribute prioritized experiences, while a central learner samples high-priority transitions for training. IMPALA [34] extends this paradigm to policy-based methods using V-trace off-policy correction to adjust for actor-learner policy discrepancy, supporting massive actor parallelization suitable for planet-scale deployments.

In large-scale IoT-Edge-Cloud systems, experience-based parallelism enables learning from massive distributed data sources. Zhang et al. [211] employed Ape-X for building demand response control, balancing thermal comfort and energy consumption across edge-cloud infrastructure, with further applications in scheduling tuning [52], mobile crowdsensing [102], and MEC placement [163]. Wang et al. [175] proposed TF-DDRL following IMPALA for IoT application scheduling, jointly minimizing response time, energy, and cost, with further applications in distributed placement [47] and adaptive resource management [174].

Experience-based methods offer unparalleled scalability, with near-linear speedup as actors increase [34, 63]. However, they require substantial infrastructure (centralized replay buffers, high-bandwidth connections to learners) and introduce complexity in managing actor-learner synchronization [34, 63, 78]. The choice between gradient-based and experience-based parallelism depends on deployment constraints: gradient-based methods suit moderate-scale edge deployments with good connectivity, while experience-based methods excel in planetary-scale IoT scenarios where data collection dominates computational costs.

4.2 Detailed Comparative Analysis

Single-agent architectures provide foundational capabilities for IoT-Edge-Cloud resource management, offering coherent global optimization when centralized control is feasible. The key advantages include unified global policies that avoid coordination challenges inherent to multi-agent approaches, mature theoretical foundations with convergence guarantees under well-established assumptions [153], implementation simplicity from managing a single policy, and well-established open-source implementations (e.g., Stable Baselines3, RLlib) that accelerate development and ensure reproducibility. However, the effectiveness of specific SARL methods varies substantially across the four system-level dimensions analyzed below.

4.2.1 ***Scalability.*** Scalability determines how well a method handles the growing number of managed resources, devices, and decision variables characteristic of large-scale IoT-Edge-Cloud deployments [15, 154].

Serial value-based methods (DQN family) exhibit the most severe scalability limitations. The discrete action space grows exponentially with decision dimensions [98, 153]: simultaneously allocating CPU, memory, and bandwidth across N edge servers yields $O(|A|^N)$ joint actions, quickly rendering Q-table enumeration intractable even for moderate-scale deployments [185]. Dueling DQN improves learning efficiency under large action sets through value-advantage decomposition [177], but does not resolve the exponential growth, limiting these methods to small-to-moderate resource pools [4, 210]. Serial policy-based methods (TRPO, PPO) improve scalability by handling continuous and high-dimensional actions without discretization, scaling more gracefully with decision dimensions [144]. PPO's lower per-update cost compared to TRPO makes it particularly practical for larger-scale edge deployments [2, 176]. Serial actor-critic methods (DDPG, TD3, SAC) achieve similar scalability through continuous control, with SAC's stochastic policy handling higher-dimensional action spaces more robustly via entropy-driven exploration [51, 157]. Parallel gradient-based methods (A3C, A2C) enhance scalability through multi-worker parallelism that naturally maps to distributed edge clusters [118, 214, 223], though gradient communication overhead limits further scaling in bandwidth-constrained edge networks. A2C's synchronous barrier additionally constrains scaling efficiency compared to A3C [109]. Parallel experience-based methods (Ape-X, IMPALA) achieve the highest scalability with near-linear speedup as actors increase [34, 63]. Actor-learner decoupling enables planetary-scale data collection across hundreds of distributed actors [47, 175]. IMPALA extends Ape-X's discrete-only applicability to both discrete and continuous actions through V-trace correction, offering broader coverage across diverse resource management tasks [174].

4.2.2 ***Convergence.*** Convergence encompasses both convergence speed (how quickly a method reaches effective policies for timely deployment) and convergence stability (how reliably it avoids training failure or catastrophic policy degradation), both critical for resource management where suboptimal policies directly translate to QoS violations and energy waste [154, 175].

Convergence speed. Serial value-based methods exhibit variable speed: vanilla DQN converges slowly due to overestimation bias [119]; Double DQN improves value accuracy through decoupled selection and evaluation [165]; Dueling DQN accelerates learning in sparse-reward settings common in edge caching and long-horizon scheduling [10, 40]; Rainbow achieves the best sample efficiency by combining six orthogonal improvements [62, 210]. Serial policy-based methods achieve faster convergence through effective per-step gradient utilization: PPO's multi-epoch clipped updates per batch provide high sample efficiency among serial methods, enabling rapid policy discovery for dynamic edge workloads [144, 169]; TRPO's conservative KL-constrained step sizes ensure progress but may slow convergence in time-sensitive deployments [143]. Serial actor-critic methods benefit from off-policy replay for high sample efficiency across all variants, though convergence characteristics differ: DDPG converges fast but may oscillate due to overestimation [98]; TD3 achieves higher asymptotic performance through delayed updates [41, 86]; SAC converges fastest in practice by reducing local optima trapping via entropy-driven exploration [51, 157]. Parallel gradient-based methods (A3C, A2C) achieve faster wall-clock convergence than serial on-policy methods through increased data throughput, though per-sample efficiency remains limited by the on-policy constraint [118, 164]. Parallel experience-based methods achieve the fastest wall-clock convergence among all categories: massive parallel data collection combined with off-policy learning produces per-sample efficiency lower than serial off-policy methods due to actor-learner staleness, but the throughput advantage dominates at scale [34, 63, 175].

Convergence stability. Value-based methods achieve reliable stability through experience replay and target networks across all variants, with theoretical convergence guarantees under standard assumptions [119, 165]. PPO's clipping mechanism prevents catastrophic policy degradation, while TRPO provides the strongest monotonic improvement guarantees via KL constraints [143, 144], both critical for production edge deployments where policy failures directly impact service availability [176]. Among actor-critic methods, DDPG exhibits significant sensitivity to hyperparameters; TD3 substantially improves stability through twin critics and target smoothing [41]; SAC achieves the most robust training through automatic temperature tuning [51]. A3C's asynchronous updates introduce stale gradients that may degrade final policy performance, while A2C's synchronous batching eliminates staleness for more consistent convergence [109, 118]. Ape-X maintains stable centralized learning through prioritized replay focusing on high-value transitions [63], while IMPALA's V-trace importance weighting ensures unbiased gradient estimation even under high actor-learner lag [34, 174].

4.2.3 ***Overhead.*** Overhead encompasses computational cost, memory footprint, communication bandwidth, and infrastructure requirements, all critical constraints for resource management at the network edge where controllers operate under stringent resource budgets [5, 81].

Serial value-based methods impose the lowest overhead: single-network forward passes enable fast inference suitable for latency-sensitive edge controllers [4, 97]. Memory requirements are moderate due to replay buffers. Rainbow increases training computation through distributional heads and prioritized replay, but inference overhead remains comparable to vanilla DQN [62, 210]. Serial policy-based methods incur moderate-to-high overhead: on-policy learning requires continuous environment interaction, preventing sample reuse [153]. TRPO's second-order optimization (Hessian-vector products) substantially increases per-update computation; PPO achieves comparable stability with first-order computation, making it the preferred choice for resource-constrained edge deployments [73, 144, 176]. Serial actor-critic methods impose moderate overhead through dual networks (actor + critic) plus target networks, increasing memory over single-network methods. DDPG has the lowest overhead with one critic; TD3 doubles critic computation through twin networks; SAC adds entropy computation and temperature optimization [41, 51]. All three benefit from replay buffers amortizing interaction cost, making them more communication-efficient than on-policy alternatives [17, 35]. Parallel gradient-based methods incur high overhead through frequent gradient communication per update across all workers, compounded by on-policy sample waste [118]. A3C avoids synchronization delays but wastes computation on stale updates; A2C eliminates staleness but introduces straggler delays where the slowest worker bottlenecks each round [109, 223]. The natural mapping to distributed edge infrastructure partially offsets communication costs by leveraging existing inter-node connectivity [214]. Parallel experience-based methods require the highest infrastructure overhead: centralized replay buffers, dedicated learner servers, and high-bandwidth actor-learner connections [34, 63]. Ape-X demands substantial memory for large-scale prioritized replay but only transfers compact experience tuples; IMPALA transfers richer trajectory data including full rollouts, increasing bandwidth requirements but reducing learner-side computation [47, 175].

4.2.4 ***Adaptability.*** Adaptability measures how effectively a method responds to the dynamic workloads, non-stationary network conditions, and evolving device availability inherent to IoT-Edge-Cloud environments [154, 162, 175].

Serial value-based methods exhibit the lowest adaptability. Fixed discrete action sets cannot adjust to changing resource granularity (e.g., when new servers are added or removed), requiring retraining or re-discretization when system configurations evolve [185]. Double DQN and Dueling DQN do not improve this limitation; Rainbow's noisy networks provide limited exploration under distribution shift but cannot fundamentally overcome the rigidity of discrete action representations [62]. Serial policy-based methods achieve moderate adaptability: direct policy parameterization enables flexible online adjustment, and on-policy learning naturally tracks distribution shifts through continuous data collection [169]. TRPO's conservative updates provide stable but slow adaptation; PPO's clipped updates enable faster re-adaptation while maintaining safety [22, 144]. Serial actor-critic methods offer moderate-to-high adaptability through off-policy replay, which enables learning from historical data under changing conditions [20, 192]. DDPG lacks explicit exploration, degrading adaptability in non-stationary edge environments; TD3's target smoothing provides moderate robustness [41, 86]; SAC's maximum entropy framework maintains persistent exploration, offering the strongest adaptability to dynamic workloads among serial methods [30, 51, 96]. Parallel gradient-based methods achieve moderate-to-high adaptability through diverse parallel exploration that improves state-action coverage across heterogeneous edge environments [164, 214]. A3C's asynchronous nature enables continuous real-time adaptation without synchronization barriers; A2C provides more coordinated adaptation but introduces latency between environment changes and policy responses [109, 223]. Parallel experience-based methods achieve the highest adaptability: massive actor parallelism provides diverse exploration across heterogeneous edge environments simultaneously [47, 175]. Ape-X's prioritized replay emphasizes surprising transitions, enabling rapid adaptation to anomalous conditions such as flash crowds or node failures [63, 211]; IMPALA's V-trace correction maintains policy accuracy under high actor-learner lag, making it well-suited for rapidly evolving large-scale IoT deployments [34, 174].

Table 3 systematically compares SARL methods across scalability, convergence, overhead, and adaptability.

Table 3. Systematic Comparison of Single-Agent DRL Methods for IoT-Edge-Cloud Continuum Resource Management

Category	**Method**	**Year**	**Action Space**	**Key Characteristics**	**Literature**	**Policy Type**	**Scalability**	**Convergence**	**Overhead**	**Adaptability**
Serial – Value-Based	DQN	2015	Discrete	Experience replay; Target network; Pioneering deep Q-learning	[4, 55, 185, 207]	Off	**Low.** Discrete actions only; action space grows exponentially with decision dimensions, limiting deployment to small-to-moderate resource pools. All variants share this fundamental constraint; Dueling DQN improves learning efficiency under large action sets but does not resolve the exponential growth	**Moderate.** *Speed:* DQN converges slowly due to overestimation bias; Double DQN improves value accuracy; Dueling DQN accelerates learning in sparse-reward settings; Rainbow achieves the best sample efficiency by combining six orthogonal improvements. *Stability:* Experience replay and target networks provide stable training across all variants; theoretical convergence guarantees hold under standard assumptions	**Low–Moderate.** Single-network forward pass enables fast inference at edge. All variants share similar low inference cost; Rainbow increases training computation through distributional heads and prioritized replay but inference overhead remains comparable to DQN	**Low.** Fixed discrete action sets cannot adapt to changing resource granularity; requires retraining or re-discretization when system configuration changes. Double DQN and Dueling DQN do not improve adaptability; Rainbow's noisy networks provide limited exploration under distribution shift
	Double DQN	2016	Discrete	Decoupled action selection/evaluation; Reduces overestimation	[19, 97, 170]					
	Dueling DQN	2016	Discrete	Value-advantage decomposition; Efficient when many actions yield similar values	[10, 40]					
	Rainbow	2018	Discrete	6 orthogonal DQN extensions; Superior sample efficiency	[27, 37, 121, 202, 210]					
Serial – Policy-Based	TRPO	2015	Cont/Disc	KL-constrained updates; Monotonic improvement; Second-order optimization	[13, 128, 131]	On	**Moderate.** Handles continuous and high-dimensional actions without discretization; scales better with decision dimensions than value-based methods. TRPO and PPO share similar scalability; PPO's lower per-update cost makes it more practical for larger-scale deployments	**Good.** *Speed:* PPO achieves fast convergence through multi-epoch clipped updates per batch, offering high sample efficiency among serial methods; TRPO's conservative KL-constrained step sizes ensure progress but may slow convergence. *Stability:* PPO clipping prevents catastrophic policy degradation; TRPO provides the strongest monotonic improvement guarantees via KL constraints	**Moderate–High.** On-policy nature requires continuous environment interaction, limiting sample reuse. TRPO incurs expensive second-order optimization (Hessian-vector products); PPO achieves comparable stability with first-order computation, substantially reducing per-update cost and making it the preferred choice for edge deployments	**Moderate.** Direct policy parameterization enables flexible online adjustment; on-policy learning tracks distribution shifts naturally through continuous data collection. TRPO's conservative updates provide stable but slow adaptation; PPO's clipped updates enable faster re-adaptation while maintaining safety against catastrophic policy degradation
	PPO	2017	Cont/Disc	Clipped surrogate objective; First-order efficiency; Production-grade	[2, 22, 73, 74, 90, 169, 176]					
Serial – Actor-Critic	DDPG	2016	Continuous	Deterministic policy; DQN-style replay for continuous control	[3, 17, 20, 48, 58]	Off	**Moderate.** Continuous control scales well with resource dimensions; off-policy replay improves data utilization. All three variants share similar scalability; SAC's stochastic policy handles higher-dimensional action spaces more robustly than DDPG's deterministic policy through entropy-driven exploration	**Good.** *Speed:* Off-policy replay enables high sample efficiency across all variants; DDPG converges fast but may oscillate due to overestimation; TD3 achieves higher asymptotic performance through delayed updates and target smoothing; SAC converges fastest in practice via entropy-driven exploration, reducing local optima trapping. *Stability:* DDPG is sensitive to hyperparameters; TD3 significantly improves stability via twin critics and target smoothing; SAC achieves the most robust training through automatic temperature tuning	**Moderate.** Dual networks (actor + critic) plus target networks increase memory and computation over single-network methods. DDPG has the lowest overhead with one critic; TD3 doubles critic computation through twin networks; SAC adds entropy computation and temperature optimization but all benefit from replay buffer amortizing interaction cost	**Moderate–High.** Off-policy replay enables learning from historical data under changing conditions. DDPG lacks explicit exploration, degrading adaptability in non-stationary environments; TD3's target smoothing provides moderate robustness; SAC's maximum entropy framework maintains persistent exploration, offering the strongest adaptability to dynamic edge workloads
	TD3	2018	Continuous	Twin critics; Delayed policy updates; Target smoothing	[35, 53, 86, 147, 192]					
	SAC	2018	Continuous	Maximum entropy framework; Auto temperature tuning; Robust	[30, 60, 96, 150, 157]					
Parallel – Gradient-Based	A3C	2016	Cont/Disc	Asynchronous workers; Advantage function; Immediate updates	[33, 76, 164, 214, 223]	On	**Moderate–High.** Multi-worker parallelism accelerates training and naturally maps to distributed edge clusters. A3C and A2C share similar scalability bounds; both are limited by gradient communication overhead in bandwidth-constrained networks, with A2C's synchronous barrier further constraining scaling efficiency	**Moderate.** *Speed:* Parallel workers increase data throughput, achieving faster wall-clock convergence than serial on-policy methods; however, on-policy nature limits per-sample efficiency compared to off-policy methods. *Stability:* A3C's asynchronous updates introduce stale gradients that may degrade convergence quality and final policy performance; A2C's synchronous batching eliminates staleness, providing more stable and consistent convergence	**High.** Frequent gradient communication per update across all workers; on-policy nature prevents sample reuse. A3C avoids synchronization overhead but wastes computation on stale updates; A2C eliminates staleness but introduces straggler delays where the slowest worker bottlenecks each round	**Moderate–High.** Diverse parallel exploration across workers improves state-action coverage in heterogeneous edge environments. A3C's asynchronous nature enables continuous real-time adaptation without synchronization barriers; A2C's synchronized updates provide more coordinated adaptation but introduce latency between environment changes and policy responses
	A2C	2016	Cont/Disc	Synchronous batching; Stable gradient aggregation; Consistency	[71, 93, 109, 156, 188]					
Parallel – Experience-Based	Ape-X	2018	Discrete	Hundreds of actors; Prioritized distributed replay	[52, 102, 163, 211]	Off	**High.** Near-linear speedup with actor count; actor-learner decoupling enables planetary-scale data collection. Ape-X scales to hundreds of actors for discrete control; IMPALA extends this to both discrete and continuous actions with V-trace correction, offering broader applicability across resource management tasks	**Good.** *Speed:* Massive parallel data collection combined with off-policy learning achieves the fastest wall-clock convergence among all categories; per-sample efficiency is lower than serial off-policy methods due to actor-learner staleness, but throughput advantage dominates. *Stability:* Ape-X uses prioritized replay to focus on high-value transitions with stable centralized learning; IMPALA employs V-trace importance weighting for unbiased gradient estimation, maintaining stability under high actor-learner lag	**High.** Both require centralized infrastructure (learner server, high-bandwidth connections). Ape-X demands substantial memory for large-scale prioritized replay buffers but only transfers compact experience tuples; IMPALA transfers richer trajectory data including full rollouts, increasing bandwidth requirements but reducing learner-side computation for trajectory collection	**High.** Massive actor parallelism provides diverse exploration across heterogeneous edge environments. Ape-X's prioritized replay emphasizes surprising transitions, enabling rapid adaptation to anomalous conditions; IMPALA's V-trace correction maintains policy accuracy under high actor-learner lag, better suited for rapidly evolving large-scale IoT deployments
	IMPALA	2018	Cont/Disc	Large-scale actors; V-trace off-policy correction	[47, 174, 175]					

4.3 Design Guidelines.

Based on the four-dimensional analysis, we provide practical guidelines for selecting SARL methods in IoT-Edge-Cloud resource management deployments.

Action space characteristics should drive the primary method selection. Value-based methods (DQN family) suit problems with purely discrete decisions of manageable cardinality, such as binary offloading [4] or server selection among a small set of candidates [55]. Policy-based methods (TRPO, PPO) handle continuous control and high-dimensional discrete problems, with PPO preferred for its balance of convergence speed and low overhead [2, 176]. Off-policy actor-critic methods (DDPG, TD3, SAC) excel when sample efficiency is critical and the action space is continuous, such as fine-grained resource allocation ratios or power control [86, 157]; SAC is recommended for non-stationary environments due to its superior adaptability [30, 96].

Deployment scale should determine the training architecture. Serial training suffices for moderate deployments (tens of edge nodes) where a single controller can observe global state [3, 185]. Gradient-based parallelism (A3C, A2C) suits edge clusters with reliable inter-node connectivity, with A2C preferred when convergence stability outweighs wall-clock speed [109, 223]. Experience-based parallelism (Ape-X, IMPALA) enables planetary-scale systems with massive data generation, with IMPALA preferred for its broader action space support [47, 175].

Resource budgets at the edge controller constrain feasible methods. Value-based methods impose the lowest inference overhead, suitable for resource-constrained IoT gateways [4]. Actor-critic methods require moderate resources for dual-network inference. Parallel experience-based methods demand substantial centralized infrastructure, typically hosted on cloud servers coordinating distributed edge actors [174, 175].

Environment dynamics should inform adaptability requirements. For relatively stable environments (e.g., fixed infrastructure with predictable diurnal workload patterns), value-based methods with periodic retraining may suffice [210]. For moderately dynamic environments (e.g., mobile edge computing with user mobility), PPO or SAC provides effective online adaptation [96, 169]. For highly dynamic large-scale environments (e.g., vehicular edge computing, UAV-assisted networks), parallel experience-based methods offer the strongest combination of scalability and adaptability [30, 175].

4.4 Multi-Agent Reinforcement Learning Architectures

While SARL offers coherent global optimization, it faces fundamental scalability barriers when state spaces grow exponentially with device count, communication latency violates real-time constraints, or heterogeneous edge domains exhibit conflicting local objectives [153, 154, 171]. MARL addresses these limitations by decomposing the resource management problem across multiple autonomous agents, each responsible for local decision-making within its domain (e.g., an edge server, a network slice, or a geographical region). We organize MARL approaches based on their coordination mechanisms. *Independent Learning Approaches* (Section 4.4.1) treat other agents as part of the environment, learning without explicit coordination, suitable for scenarios with weak inter-agent dependencies. *Centralized Training with Decentralized Execution (CTDE) Approaches* (Section 4.4.2) leverage global information during training to learn coordinated policies while maintaining distributed execution, the predominant paradigm for edge computing where training can occur in resource-rich cloud environments while execution must be distributed across edge nodes.

4.4.1 ***Independent Learning Approaches.***

Independent learning approaches apply single-agent DRL methods to each agent independently, treating other agents' actions and policies as part of stochastic environment dynamics. Each agent i learns its own policy $\pi_i(a_i|o_i)$ based solely on local observations and rewards, without explicit knowledge of other agents. This paradigm offers extreme simplicity and scalability, as agents require no inter-agent communication and can be trained in parallel. We distinguish three categories: *Independent Value-Based Methods*, *Independent Policy-Based Methods*, and *Independent Actor-Critic Methods*.

Independent Value-Based Methods. Independent value-based methods learn separate state-action value functions $Q_i(o_i, a_i)$ for each agent, with policies derived through independent value maximization. Each agent treats other agents' behaviors as stochastic environment dynamics, inheriting the discrete action space suitability of value-based methods. Independent DQN (IDQN) is the predominant approach in this category.

IDQN [155] applies DQN to multi-agent scenarios where each agent i maintains its own Q-network $Q_{\theta_i}(o_i, a_i)$ with independent experience replay buffer $\mathcal{D}_i$. This fully decentralized approach achieves extreme scalability, as adding new agents requires no modification to existing agents' training processes.

In IoT-Edge-Cloud resource management, independent value-based methods suit scenarios with loose coupling between edge domains. Cui et al. [24] employed IDQN to balance throughput and power consumption in UAV network resource allocation through decentralized learning, with further applications in Non-Orthogonal Multiple Access (NOMA)-enabled MEC [178] and network slice embedding [31].

The effectiveness of independent value-based methods is limited by discrete action spaces. Continuous resource allocation requires discretization, introducing approximation errors [15]. Furthermore, deep Q-functions may insufficiently capture complex state-action relationships in high-dimensional observation spaces [82, 116]. These limitations motivate independent policy gradient methods.

Independent Policy-Based Methods. Independent policy-based methods extend policy gradient methods to multi-agent settings by maintaining separate policies for each agent. Each agent directly optimizes its policy through gradient ascent on expected returns using local trajectories. This approach naturally handles continuous action spaces, suitable for scenarios requiring precise control. Independent PPO (IPPO) is the predominant approach due to its training stability.

IPPO [29] extends PPO's clipped surrogate objective to multi-agent settings. Each agent independently applies PPO updates, maintaining its own policy π_{θ_i}, old policy $\pi_{\theta_{i,old}}$, and critic network $V_{\psi_i}(o_i)$ for advantage estimation, all computed from local trajectories. IPPO inherits PPO's training stability while achieving multi-agent scalability.

Lin et al. [100] employed IPPO to maximize computing energy efficiency in vehicular edge computing, enabling each vehicle to autonomously learn offloading policies through fully distributed training. Further IPPO applications address time-sensitive IoT resource allocation [103], decentralized task offloading [132], wireless-powered MEC [105], satellite-terrestrial networks [196], and multi-agent task scheduling [75].

Independent policy gradient methods achieve continuous control but face training instability in strongly-coupled scenarios [108]. Non-stationary environments also cause oscillatory behavior when agents rapidly adjust policies [126]. While effective in weakly-coupled scenarios, these methods lack mechanisms to explicitly model other agents.

Independent Actor-Critic Methods. Independent actor-critic methods maintain separate actor-critic pairs for each agent, where agent i independently learns both a policy $\pi_{\theta_i}(a_i|o_i)$ (actor) and value function $Q_{\psi_i}(o_i, a_i)$ (critic). This approach combines continuous action capability with sample efficiency through off-policy learning and experience replay. Independent DDPG (IDDPG) serves as the primary approach in this category.

IDDPG [108] applies DDPG to multi-agent scenarios where each agent maintains its own actor network, critic network, and corresponding target networks. Each agent independently maintains an experience replay buffer $\mathcal{D}_i$ and updates networks through independent gradient descent. This achieves deterministic policy optimization without inter-agent coordination.

Zheng et al. [219] employed IDDPG to minimize AoI in wireless-powered IoT networks, with further applications in digital twin-enabled Internet of Vehicles [45] and multi-aerial base station deployment [57].

The effectiveness of independent actor-critic methods is limited by environment non-stationarity from concurrent policy updates [61]. When agents' policies evolve simultaneously, critic value estimation becomes unreliable as environment dynamics shift, potentially causing training instability or convergence to suboptimal equilibria.

4.4.2 *Centralized Training with Decentralized Execution (CTDE) Approaches*.
Independent learning approaches struggle under strong inter-agent dependencies, as agents treat each other's evolving policies as environment dynamics, leading to non-stationarity and suboptimal coordination [108, 126]. CTDE addresses this by exploiting a key asymmetry in IoT-Edge-Cloud deployments: training leverages global information in resource-rich cloud environments, while execution remains distributed across edge nodes using only local observations. We distinguish two CTDE paradigms: *Value Decomposition Methods* factorize global value functions for decentralized discrete action selection, while *Centralized Critic Methods* employ critics with global information to guide distributed actors for both discrete and continuous control.

Value Decomposition Methods. Value decomposition methods address coordinated discrete action selection while maintaining decentralized execution by factorizing global Q-functions into local components. The core idea is learning a joint action-value function $Q_{tot}(\mathbf{o}, \mathbf{a})$ that decomposes into agent-specific utilities $Q_i(o_i, a_i)$ under structural constraints that guarantee global optimal actions can be obtained through independent local maximization.

Value Decomposition Networks (VDN) [152] achieves this through additive decomposition: $Q_{tot}(\mathbf{o}, \mathbf{a}) = \sum_{i=1}^{n} Q_i(o_i, a_i)$. VDN's simplicity enables efficient training but imposes restrictive assumptions that may fail under complex non-linear agent interactions. QMIX [137] relaxes this constraint through monotonic value factorization using a mixing network while maintaining $\frac{\partial Q_{tot}}{\partial Q_i} \geq 0$, enabling more expressive value functions through hypernetworks conditioned on global state.

In edge computing scenarios, value decomposition methods have proven effective for discrete resource allocation. Raivi and Moh [136] employed VDN to minimize energy consumption and delay in UAV-enabled IoT for post-disaster data aggregation and computation offloading, with further VDN applications in digital twin edge networks [194], network slicing [142], and integrated ground-air-space networks [49]. Yu et al. [200] developed a QMIX-based delay minimization algorithm for heterogeneous UAV-swarm-enabled aerial edge computing, with further QMIX applications in network slicing [72], UAV-assisted caching [161], diffusion-based resource management [124], and vehicular task offloading [50].

Value decomposition methods provide decentralized execution guarantees through the Individual-Global-Max principle [148] and communication-free scalability [152]. However, their structural constraints and limitation to discrete actions restrict applicability in IoT-Edge-Cloud deployments with intricate inter-dependencies [166].

Centralized Critic Methods. Centralized critic methods implement CTDE by employing critics that observe global information during training to guide distributed actors, eliminating non-stationarity by directly modeling other agents' policies. This category supports both discrete and continuous action spaces.

Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [108] pioneered CTDE actor-critic for continuous control. Each agent maintains a deterministic policy $\mu_{\theta_i}(o_i)$ and a centralized critic $Q_{\phi_i}(\mathbf{o}, \mathbf{a})$ observing global state during training, while execution remains decentralized. Counterfactual Multi-Agent (COMA) [39] addresses credit assignment through counterfactual advantages, measuring each agent's marginal contribution, particularly effective for discrete action spaces. Multi-Agent Proximal Policy Optimization (MAPPO) [199] extends PPO through centralized value functions for advantage estimation, combining training stability with CTDE coordination.

Applications of centralized critic methods span multiple domains. He et al. [59] employed MADDPG to jointly maximize cache hit ratio and minimize traffic load in cloud-to-edge proactive caching, with further applications in UAV-assisted vehicular networks [127], ISAC-aided 6G V2X [66], and dynamic slice resource allocation [140]. Subhan et al. [149] developed a COMA-based approach to maximize bandwidth in 360° video streaming, with further applications in task migration [101], intelligent road networks [67], and vehicular edge computing [25]. Yao et al. [193] proposed MAPPO-based delay minimization in crowd-edge computing under partial observability, with further applications in Graph Neural Network (GNN)-based task placement [94], semantic-aware resource optimization [70], cooperative UAV allocation [77], Kubernetes scheduling [68], and small-cell MEC offloading [88].

Centralized critic methods provide maximum flexibility, supporting both continuous and discrete action spaces through different algorithmic approaches. However, compared to value decomposition methods, they typically exhibit higher sample complexity and greater sensitivity to hyperparameters [126].

4.5 Detailed Comparative Analysis

Multi-agent architectures provide essential capabilities for managing large-scale, heterogeneous IoT-Edge-Cloud deployments where centralized control becomes infeasible. The key advantages include distributed decision-making that scales naturally with system size, avoiding bottlenecks inherent to centralized control [171]; natural alignment with the inherently distributed IoT-Edge-Cloud infrastructure, where autonomous edge nodes make local decisions with limited global visibility [154]; and CTDE approaches that provide coordinated global optimization while maintaining communication-free decentralized execution [108, 137]. However, the effectiveness of specific MARL methods varies substantially across the four system-level dimensions analyzed below.

4.5.1 ***Scalability.*** Scalability in MARL encompasses both the number of agents a system can accommodate and per-agent decision granularity, both critical as IoT-Edge-Cloud deployments scale to hundreds or thousands of autonomous edge nodes [15, 154].

Independent methods (IDQN, IPPO, IDDPG) achieve the highest scalability among MARL approaches. Adding agents requires no modification to existing agents' training processes, enabling linear scaling with agent count [155, 199]. This makes independent methods particularly suited for large-scale edge deployments where each edge server operates autonomously [24, 100]. Among independent methods, IDQN is limited to discrete actions, constraining per-agent decision granularity for fine-grained resource allocation, while IPPO and IDDPG handle continuous actions, enabling finer resource control as agent count grows [103, 219]. CTDE value decomposition methods (VDN, QMIX) maintain communication-free decentralized execution that scales well with agent count, but training requires global state for mixing networks, creating centralized bottlenecks [137, 152]. VDN's simple additive structure scales more efficiently than QMIX's hypernetwork-based mixing as agent count increases [136, 200]. CTDE centralized critic methods (MADDPG, COMA, MAPPO) exhibit the most constrained training scalability: centralized critics grow with joint observation-action dimensionality, scaling quadratically with agent count for MADDPG and COMA [39, 108]. MAPPO's centralized value function scales more gracefully through shared observation processing, partially alleviating this bottleneck [193, 199].

4.5.2 ***Convergence.*** Convergence in multi-agent settings faces unique challenges absent in SARL, particularly non-stationarity from concurrent policy updates and credit assignment difficulty when rewards are sparse or global [61, 126].

Convergence speed. Independent methods exhibit the slowest convergence due to non-stationarity: each agent treats other agents' evolving policies as environment dynamics, creating moving targets that slow learning [61]. IDQN additionally inherits DQN's overestimation-related slow convergence [24, 178]; IPPO converges faster per agent through PPO's multi-epoch updates [100, 103]; IDDPG benefits from off-policy sample efficiency but deterministic policies may oscillate [219]. CTDE methods achieve substantially faster convergence by eliminating non-stationarity during training through centralized information access. VDN's simple additive structure converges faster but with lower asymptotic quality; QMIX's monotonic mixing achieves better final performance at the cost of slower convergence [137, 200]. Among centralized critic methods, MADDPG benefits from off-policy efficiency [59, 127]; COMA's counterfactual baselines provide principled credit assignment that accelerates learning when global rewards dominate [101, 149]; MAPPO achieves surprisingly strong convergence through centralized value estimation combined with PPO's sample-efficient updates [94, 193, 199].

Convergence stability. Independent methods face the most severe stability challenges: non-stationarity from concurrent updates violates the Markov assumption, frequently leading to convergence to suboptimal equilibria [61]. IPPO's clipping provides per-agent stability but cannot prevent system-level oscillations in strongly-coupled scenarios [126, 132]; IDDPG compounds this with DDPG's inherent hyperparameter sensitivity [219]. CTDE value decomposition methods provide strong structural stability: the Individual-Global-Max (IGM) principle guarantees consistent decentralized execution, and structural constraints (additivity in VDN, monotonicity in QMIX) regularize training [72, 137, 148]. Among centralized critic methods, MADDPG inherits DDPG's hyperparameter sensitivity [59]; COMA's counterfactual baselines stabilize multi-agent credit assignment [39, 67]; MAPPO's clipping mechanism ensures stable policy updates across all agents, providing the most robust convergence among centralized critic methods [77, 193].

4.5.3 ***Overhead.*** Overhead in MARL spans per-agent computation and memory, inter-agent communication, and centralized training infrastructure, all amplified by the distributed nature of IoT-Edge-Cloud deployments where edge nodes operate under stringent resource constraints [5, 81].

Independent methods impose the lowest overhead: no inter-agent communication during training or execution, with per-agent costs matching their single-agent counterparts [155, 199]. IDQN and IPPO have minimal per-agent footprints; IDDPG's dual networks (actor + critic + targets) increase per-agent memory compared to IDQN and IPPO [45, 219]. CTDE value decomposition methods eliminate runtime communication through decentralized execution, but training requires global state collection and mixing network computation [137, 152]. VDN incurs minimal mixing overhead (summation), while QMIX requires hypernetwork forward passes conditioned on global state, increasing training computation [124, 200]. CTDE centralized critic methods impose the highest overhead among MARL approaches: communication-free execution is maintained, but training requires global state exchange for centralized critics, creating bandwidth demands in distributed edge deployments [108]. MADDPG's shared replay buffer adds substantial memory overhead; COMA's counterfactual computation enumerates all per-agent actions, with cost scaling with action space size [39, 149]; MAPPO balances overhead through efficient centralized value estimation without per-agent action enumeration [88, 193].

4.5.4 ***Adaptability.*** Adaptability in MARL encompasses both per-agent responsiveness to local environment changes and collective adaptation to system-wide shifts, the latter being a unique challenge in multi-agent settings where individual adaptation may conflict with global coordination [162, 175].

Independent methods exhibit limited collective adaptability despite per-agent responsiveness. IDQN's replay buffers enable adaptation to local changes but lack coordination for system-wide shifts such as cascading workload migrations or correlated network failures [24, 178]. IPPO tracks local distribution shifts through on-policy learning with clipped updates, preventing catastrophic degradation [100, 103]. IDDPG's off-policy replay provides historical learning capability but deterministic policies lack exploration for adapting to non-stationary dynamics from other agents [219]. CTDE value decomposition methods provide moderate adaptability: decentralized execution enables local responsiveness, but fixed structural constraints (VDN's additivity, QMIX's monotonicity) limit expressiveness for complex non-linear inter-dependencies in IoT-Edge-Cloud deployments, such as cascading workload effects and heterogeneous resource correlations [136, 166, 200]. CTDE centralized critic methods achieve the highest adaptability by modeling inter-agent interactions through centralized critics, enabling coordinated adaptation to system-wide changes [108]. MADDPG adapts through off-policy replay but lacks exploration [59, 66]; COMA's credit assignment enables targeted per-agent adaptation by identifying individual contributions to collective outcomes [25, 149]; MAPPO combines coordinated adaptation with PPO's on-policy tracking of distribution shifts, offering the best balance for dynamic IoT-Edge-Cloud environments such as vehicular edge computing and UAV-assisted networks [68, 77, 193].

Table 4 systematically compares MARL methods across scalability, convergence, overhead, and adaptability.

Table 4. Systematic Comparison of Multi-Agent DRL Methods for IoT-Edge-Cloud Continuum Resource Management

Category	**Method**	**Year**	**Action Space**	**Key Characteristics**	**Literature**	**Coordination**	**Scalability**	**Convergence**	**Overhead**	**Adaptability**
Independent – Value-Based	IDQN	2017	Discrete	Per-agent replay; Local observations only; DQN extension	[24, 31, 178]	Implicit	**High.** Adding agents requires no modification to existing agents; scales linearly with agent count. However, discrete action spaces limit per-agent decision granularity in fine-grained resource allocation scenarios	**Low.** *Speed:* Per-agent sample efficiency inherits DQN limitations; concurrent policy changes slow convergence. *Stability:* Non-stationarity from other agents' evolving policies violates the Markov assumption, causing training instability and frequent convergence to suboptimal equilibria	**Low.** No inter-agent communication; per-agent replay buffers have moderate memory cost; fully decentralized training and execution with minimal infrastructure requirements	**Low–Moderate.** Per-agent replay enables some adaptation to local changes; however, lack of coordination means agents cannot collectively adapt to system-wide shifts such as cascading workload migrations or correlated network failures
Independent – Policy-Based	IPPO	2020	Cont/Disc	Clipped objective; Independent critics; PPO extension	[75, 100, 103, 105, 132, 196]	Implicit	**High.** Inherits IDQN's linear scaling; additionally handles continuous actions without discretization, enabling finer-grained resource allocation as agent count grows	**Moderate.** *Speed:* PPO's multi-epoch updates provide faster per-agent convergence than IDQN; on-policy nature limits sample reuse but ensures fresh gradient signals. *Stability:* Clipping mechanism provides per-agent stability; however, non-stationarity from concurrent policy updates may cause oscillatory behavior in strongly-coupled scenarios	**Low.** No inter-agent communication; on-policy nature requires continuous environment interaction per agent but avoids replay buffer memory; fully decentralized with minimal infrastructure	**Moderate.** On-policy learning tracks local distribution shifts naturally through continuous data collection; clipped updates prevent catastrophic degradation; but lacks inter-agent coordination for system-wide adaptation
Independent – Actor-Critic	IDDPG	2017	Continuous	Separate actor-critic pairs; Independent replay; Deterministic policy	[45, 57, 219]	Implicit	**High.** Linear scaling with agent count; continuous control handles fine-grained resource allocation. Dual-network architecture per agent increases per-agent model complexity compared to IDQN and IPPO	**Low–Moderate.** *Speed:* Off-policy replay provides better sample efficiency than IPPO; deterministic policy enables efficient gradient computation. *Stability:* Inherits DDPG's overestimation bias and hyperparameter sensitivity, compounded by multi-agent non-stationarity; critic value estimation becomes unreliable as other agents' policies evolve	**Low–Moderate.** No inter-agent communication; per-agent replay buffers and dual networks (actor + critic + targets) increase memory footprint compared to IDQN and IPPO	**Moderate.** Off-policy replay enables learning from historical data under changing local conditions; deterministic policy lacks exploration, limiting adaptation to non-stationary dynamics from other agents' evolving behaviors
CTDE – Value Decomp.	VDN	2018	Discrete	Additive factorization; Decentralized argmax; Linear mixing	[49, 136, 142, 194]	Explicit; Value mixing	**Moderate–High.** Communication-free decentralized execution scales well with agent count. Training requires global state for mixing networks, creating centralized bottlenecks. VDN's additive structure scales more efficiently than QMIX's hypernetwork-based mixing as agent count increases	**Moderate–High.** *Speed:* Centralized training with shared replay enables efficient learning; VDN's simple additive structure converges faster but with lower asymptotic quality; QMIX's monotonic mixing achieves better final performance at the cost of slower convergence. *Stability:* the Individual-Global-Max (IGM) principle guarantees consistent decentralized execution; centralized training eliminates non-stationarity; structural constraints regularize training	**Moderate.** Communication-free execution eliminates runtime overhead; training requires global state collection and mixing network computation. VDN incurs minimal mixing overhead (summation); QMIX requires hypernetwork forward passes conditioned on global state, increasing training computation	**Moderate.** Decentralized execution enables local responsiveness to per-agent environment changes; however, fixed structural constraints (VDN's additivity, QMIX's monotonicity) limit expressiveness for complex non-linear inter-dependencies such as cascading workload effects and heterogeneous resource correlations in IoT-Edge-Cloud deployments
	QMIX	2018	Discrete	Monotonic mixing; Hypernetworks; Non-linear factorization	[50, 72, 124, 161, 200]					
CTDE – Centralized Critic	MADDPG	2017	Continuous	Deterministic policy; Shared replay buffer; DDPG extension	[59, 66, 127, 140]	Explicit; Centralized critic	**Moderate.** Decentralized execution scales well; training scalability is limited by centralized critics that grow with joint observation-action dimensionality. MADDPG and COMA critics scale quadratically with agent count; MAPPO's centralized value function scales more gracefully through shared observation processing	**Good.** *Speed:* Centralized critics eliminate non-stationarity during training, enabling faster convergence than independent methods; MAPPO achieves surprisingly strong performance through centralized value estimation with PPO's sample-efficient updates. *Stability:* MADDPG inherits DDPG's hyperparameter sensitivity; COMA's counterfactual baselines provide principled credit assignment; MAPPO's clipping ensures stable updates across all agents	**Moderate–High.** Communication-free execution; training requires global state exchange for centralized critics, creating bandwidth demands in distributed edge deployments. MADDPG's shared replay buffer adds memory overhead; COMA's counterfactual computation enumerates all per-agent actions; MAPPO balances overhead through efficient centralized value estimation	**Moderate–High.** Centralized critics model inter-agent interactions, enabling coordinated adaptation to system-wide changes. MADDPG adapts through off-policy replay but lacks exploration; COMA's credit assignment enables targeted per-agent adaptation; MAPPO combines coordinated adaptation with PPO's on-policy tracking of distribution shifts, offering the best balance for dynamic environments
	COMA	2018	Discrete	Counterfactual advantages; Action enumeration; Explicit credit assignment	[25, 67, 101, 149]					
	MAPPO	2022	Cont/Disc	Clipped objective; Variance reduction; PPO extension	[68, 70, 77, 88, 94, 193]					

4.6 Design Guidelines.

Based on the four-dimensional analysis, we provide practical guidelines for selecting MARL methods in IoT-Edge-Cloud resource management deployments.

Inter-agent dependency strength should drive coordination mechanism selection. Independent learning (IDQN, IPPO, IDDPG) suits scenarios with weak inter-agent dependencies and isolated resource pools, such as geographically separated edge clusters managing independent IoT device groups [24, 100, 219]. CTDE approaches are essential when agents share resources, experience cascading effects, or must optimize global objectives, such as cooperative offloading across overlapping coverage areas [59, 193, 200].

System scale dictates coordination feasibility. Independent methods scale linearly to thousands of agents, suited for large-scale IoT deployments with minimal inter-node coupling [103, 178]. CTDE value decomposition methods (VDN, QMIX) scale to moderate agent counts with communication-free execution, suitable for cooperative UAV swarms or regional edge clusters [136, 200]. CTDE centralized critic methods (MADDPG, COMA, MAPPO) scale to smaller agent counts before training communication costs dominate, best suited for coordinated resource management within individual edge domains [59, 88, 193].

Action space characteristics inform method selection within each coordination category. Discrete action spaces (binary offloading, server selection) favor value decomposition methods (VDN, QMIX) for their decentralized execution guarantees, or IDQN/COMA for simpler deployment [31, 136, 149]. Continuous control (power allocation, bandwidth sharing) requires actor-critic methods: IDDPG or MADDPG for fully continuous spaces [59, 219], IPPO or MAPPO when both discrete and continuous decisions coexist [70, 103].

Environment dynamics determine adaptability requirements. For moderately dynamic environments with localized changes, independent methods with per-agent adaptation suffice [100, 132]. For highly dynamic environments with inter-dependent workload patterns (e.g., vehicular edge computing, coordinated UAV networks), CTDE centralized critic methods, particularly MAPPO, provide the strongest coordinated adaptability [68, 77, 193].

4.7 SARL vs MARL

Table 5 provides a detailed literature-level comparison of DRL methods for resource management in the IoT-Edge-Cloud continuum under the standard training paradigm, organized by control architecture categories. For each work, we report the DRL algorithm used, deployment scenario (e.g., MEC, Vehicular Edge Computing (VEC), UAV-MEC), network type (e.g., cellular, V2X, satellite, wired/Data Center (DC)), decision tasks addressed (offloading, scheduling, allocation, caching, migration, and trajectory planning), task dependency model (independent or DAG-structured dependent), optimization objectives pursued (latency, energy, cost, throughput, age of information, resource utilization, and success rate), objective type (single- or multi-objective), whether mobility is considered, key system constraints (e.g., deadline, energy budget, QoS), scalability (whether the work supports scaling to larger numbers of infrastructure nodes and end devices), and evaluation methodology (simulation, trace-driven, or real-world testbed).

Table 5. Literature-Level Comparison of DRL Methods for Resource Management in IoT-Edge-Cloud Continuum (Standard Training Paradigm)

Reference	Method	Scenario	Network	Decision Task							Task Model	Optimization Objectives								Obj. Type	Mobility	Constraints	Scalability	Evaluation
				Offload	*Schedule*	*Allocate*	*Cache*	*Migrate*	*Traj.*	*Other*		*Latency*	*Energy*	*Cost*	*Thput.*	*AoI*	*Res.Util.*	*Succ.R.*	*Other*					
SARL – Serial Value-Based																								
Xiong et al. [185]	DQN	MEC	Cellular	✓	✓	✓	–	–	–	–	Indep.	✓	–	–	–	–	✓	–	–	Multi	–	–	–	Simulation
Hao et al. [55]	DQN	IIoT-MEC	WiFi/Cell.	–	✓	✓	–	✓	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	–	–	–	Simulation
Ale et al. [4]	DQN	MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	–	Deadline	–	Simulation
Zhang et al. [207]	DQN	MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	✓	✓	–	–	–	–	–	✓	Multi	–	Energy budget	–	Simulation
Liao et al. [97]	Double DQN	MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	✓	–	–	Simulation
Chen et al. [19]	Double DQN	UAV-MEC	Air-Ground	✓	–	–	–	–	–	–	Indep.	–	✓	–	–	✓	–	–	–	Multi	✓	Bandwidth	–	Simulation
Wang et al. [170]	Double DQN	Edge-Cloud	WiFi/Cell.	–	–	–	–	✓	–	–	Indep.	–	–	–	✓	–	–	–	✓	Multi	–	Reliability	–	Simulation
Birhanu T. et al. [10]	Dueling DQN	VEC	V2X	✓	–	–	–	–	–	–	Indep.	–	–	–	–	✓	–	–	–	Single	✓	Queue stability	–	Simulation
Fu et al. [40]	Dueling DQN	Edge-Cloud	WiFi/Cell.	–	–	✓	–	✓	–	–	DAG	✓	–	–	–	–	✓	–	–	Multi	–	Deadline	–	Testbed
Zhang et al. [210]	Rainbow	IIoT-MEC	WiFi/Cell.	–	✓	–	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	–	Deadline	–	Simulation
Yuan et al. [202]	Rainbow	VEC	Cellular	–	–	✓	✓	–	–	–	Indep.	–	✓	–	–	✓	–	–	–	Multi	✓	Deadline	✓	Simulation
Farimani et al. [37]	Rainbow	VEC	V2X	✓	–	–	–	–	–	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	✓	Deadline	–	Trace-driven
Nguyen et al. [121]	Rainbow	Cloud	Wired/DC	–	–	✓	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	–	–	–	Simulation
Darchini et al. [27]	Rainbow	UAV-MEC	Air-Ground	✓	✓	✓	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	✓	Energy budget; Kinematic	–	Simulation
SARL – Serial Policy-Based																								
Priyadarshni et al. [131]	TRPO	MEC	Cellular	✓	–	–	–	–	–	–	DAG	✓	✓	–	–	–	–	–	–	Multi	–	Deadline; Energy budget	–	Simulation
Peng et al. [128]	TRPO	VEC	V2X	–	–	✓	–	–	–	–	Indep.	–	✓	–	–	✓	–	–	–	Multi	✓	Energy budget; Interference	–	Simulation
Cai et al. [13]	TRPO	MEC	Cellular	–	–	✓	–	–	–	–	Indep.	✓	–	–	✓	–	–	–	–	Single	–	QoS; Energy budget	–	Simulation
Wang et al. [176]	PPO	Edge-Cloud	WiFi/Cell.	–	✓	–	–	–	–	–	DAG	✓	–	–	–	–	✓	–	–	Multi	–	–	–	Testbed
Wang et al. [169]	PPO	MEC	Cellular	✓	–	–	–	–	–	–	DAG	✓	–	–	–	–	–	–	–	Single	✓	–	–	Simulation
Chen et al. [22]	PPO	MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	✓	–	–	✓	–	–	–	–	Multi	–	Deadline; Bandwidth	–	Simulation
Jin et al. [74]	PPO	Cloud	Wired/DC	–	✓	–	–	–	–	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	–	Deadline; Energy budget	✓	Simulation
Li et al. [90]	PPO	Cloud	Wired/DC	–	–	–	–	✓	–	–	Indep.	–	–	–	✓	–	–	–	–	Single	–	Reliability	–	Trace-driven
Jimenez et al. [73]	PPO	Edge-Cloud	WiFi/Cell.	–	–	✓	–	–	–	–	Indep.	✓	–	✓	–	–	✓	–	–	Multi	–	QoS	–	Trace-driven
Agarwal et al. [2]	PPO	Edge-Cloud	WiFi/Cell.	–	✓	✓	–	–	–	–	Indep.	✓	–	✓	✓	–	–	–	–	Multi	–	QoS	–	Testbed
SARL – Serial Actor-Critic																								
Ale et al. [3]	DDPG	MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	–	Deadline	–	Simulation
Chen et al. [20]	DDPG	IIoT-MEC	Cellular	–	–	✓	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	–	Energy budget	–	Simulation
Chen et al. [17]	DDPG	MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	–	Deadline	–	Simulation
Gu et al. [48]	DDPG	MEC	Cellular	–	✓	✓	–	–	–	–	Indep.	–	–	–	✓	–	–	–	–	Single	✓	QoS	–	Testbed
He et al. [58]	DDPG	MEC	Cellular	✓	–	✓	✓	–	–	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	–	Deadline	–	Simulation
Li et al. [86]	TD3	VEC	V2X	–	–	✓	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	✓	Energy budget	–	Simulation
Hao et al. [53]	TD3	UAV-MEC	Air-Ground	✓	–	✓	–	–	✓	–	Indep.	–	–	–	–	–	–	✓	–	Single	✓	Deadline; Energy budget	–	Simulation
Sohaib et al. [147]	TD3	VEC	V2X	–	–	✓	–	–	–	–	Indep.	–	✓	–	–	–	–	–	–	Single	✓	Deadline; Energy budget	–	Simulation
Fan et al. [35]	TD3	Edge-Cloud	WiFi/Cell.	✓	–	–	–	–	–	–	DAG	–	–	–	–	–	✓	–	–	Single	–	Deadline	✓	Simulation
Yao et al. [192]	TD3	VEC	V2X	✓	–	✓	–	–	–	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	✓	–	✓	Simulation
Tang et al. [157]	SAC	MEC	WiFi/Cell.	–	–	✓	–	–	–	–	Indep.	–	✓	✓	–	–	–	–	–	Multi	–	–	✓	Simulation
Liang et al. [96]	SAC	VEC	V2X	✓	–	✓	–	–	–	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	✓	Deadline	–	Simulation
Sun et al. [150]	SAC	MEC	Cellular	✓	–	–	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	–	–	✓	Simulation
Heidarpour et al. [60]	SAC	MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	–	–	–	–	–	–	–	✓	Single	–	Bandwidth	–	Simulation
Deng et al. [30]	SAC	UAV-MEC	Air-Ground	✓	✓	✓	–	–	✓	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	✓	Energy budget	✓	Simulation
SARL – Parallel Gradient-Based																								

Continued on next page

Table 5. (continued)

Reference	Algorithm	Scenario	Network	Decision Task							Task Model	Optimization Objectives								Obj. Type	Mobility	Constraints	Scalability	Evaluation
				Offload	*Schedule*	*Allocate*	*Cache*	*Migrate*	*Traj.*	*Other*		*Latency*	*Energy*	*Cost*	*Thput.*	*AoI*	*Res.Util.*	*Succ.R.*	*Other*					
Zou et al. [223]	A3C	MEC	Cellular	✓	✓	–	–	–	–	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	–	Deadline	–	Simulation
Zhang et al. [214]	A3C	IIoT-MEC	WiFi/Cell.	✓	✓	✓	–	–	–	–	Indep.	✓	–	✓	✓	–	✓	–	–	Multi	–	Deadline	✓	Trace-driven
Tuli et al. [164]	A3C	Edge-Cloud	WiFi/Cell.	–	✓	✓	–	✓	–	–	Indep.	✓	✓	✓	–	–	–	–	–	Multi	✓	–	✓	Trace-driven
Ju et al. [76]	A3C	VEC	Cellular	✓	–	✓	–	–	–	–	Indep.	–	✓	–	–	–	–	–	–	Single	✓	Deadline; Security	–	Simulation
Du et al. [33]	A3C	MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	–	–	✓	–	–	–	–	–	Single	–	–	–	Simulation
Lu et al. [109]	A2C	Edge-Cloud	WiFi/Cell.	–	✓	✓	–	–	–	–	Indep.	–	–	–	–	–	✓	–	–	Single	–	Deadline	✓	Trace-driven
Tang et al. [156]	A2C	MEC	Cellular	✓	–	✓	✓	–	–	–	Indep.	✓	–	–	–	–	✓	–	–	Multi	✓	Deadline; Energy budget	✓	Simulation
Li et al. [93]	A2C	MEC	Cellular	✓	✓	–	✓	–	–	–	Indep.	–	–	✓	–	–	–	–	–	Single	–	Deadline	✓	Simulation
Jiang et al. [71]	A2C	UAV-MEC	Air-Ground	✓	–	✓	–	–	✓	–	Indep.	–	✓	–	–	–	–	–	–	Single	–	–	✓	Simulation
Yang et al. [188]	A2C	MEC	Cellular	–	✓	✓	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	–	–	✓	Trace-driven
SARL – Parallel Experience-Based																								
Zhang et al. [211]	Ape-X	Edge-Cloud	WiFi/Cell.	–	–	✓	–	–	–	✓	Indep.	–	✓	–	–	–	–	–	✓	Multi	–	Energy budget	✓	Simulation
Han et al. [52]	Ape-X	Edge-Cloud	WiFi/Cell.	–	✓	–	–	–	–	–	DAG	✓	–	–	–	–	–	–	–	Single	–	–	–	Trace-driven
Liu et al. [102]	Ape-X	UAV-MEC	Air-Ground	–	–	–	–	–	✓	–	Indep.	–	✓	–	–	–	–	–	✓	Multi	✓	Energy budget	✓	Simulation
Torres-Pérez et al. [163]	Ape-X	MEC	Cellular	–	–	✓	–	✓	–	–	Indep.	–	–	–	–	–	✓	–	–	Single	–	QoS	✓	Simulation
Wang et al. [175]	IMPALA	Edge-Cloud	WiFi/Cell.	–	✓	✓	–	–	–	–	DAG	✓	✓	✓	–	–	–	–	–	Multi	–	Deadline	–	Testbed
Goudarzi et al. [47]	IMPALA	Edge-Cloud	WiFi/Cell.	–	–	–	–	✓	–	–	DAG	✓	✓	–	–	–	–	–	–	Multi	–	Deadline	–	Testbed
Wang et al. [174]	IMPALA	Edge-Cloud	WiFi/Cell.	–	✓	✓	–	–	–	–	DAG	✓	✓	–	–	–	–	–	–	Multi	–	Deadline	✓	Testbed
MARL – Independent Value-Based																								
Cui et al. [24]	IDQN	UAV-MEC	Air-Ground	–	–	✓	–	–	–	–	Indep.	–	✓	–	✓	–	–	–	–	Multi	✓	Interference	✓	Simulation
Waqar et al. [178]	IDQN	MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	–	–	–	✓	–	–	–	–	Single	–	–	✓	Simulation
Doanis et al. [31]	IDQN	MEC	Cellular	–	–	–	–	✓	–	–	DAG	✓	–	✓	–	–	✓	–	–	Multi	–	QoS	✓	Trace-driven
MARL – Independent Policy-Based																								
Lin et al. [100]	IPPO	VEC	V2X	✓	–	✓	–	–	–	–	Indep.	–	✓	–	✓	–	–	–	–	Multi	✓	Deadline; Energy budget	✓	Simulation
Liu et al. [103]	IPPO	MEC	WiFi/Cell.	–	–	✓	–	–	–	–	Indep.	–	✓	–	✓	–	–	–	–	Multi	–	Energy budget; Interference	✓	Simulation
Qin et al. [132]	IPPO	MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	✓	Deadline	✓	Simulation
Liu et al. [105]	IPPO	MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	–	✓	–	–	–	–	–	–	Single	–	Energy budget; Deadline	✓	Simulation
Yin et al. [196]	IPPO	MEC	Satellite	–	✓	✓	✓	–	–	–	Indep.	–	–	–	✓	–	–	–	✓	Multi	✓	QoS	✓	Simulation
Jin et al. [75]	IPPO	MEC	Cellular	✓	✓	–	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	–	Deadline	✓	Simulation
MARL – Independent Actor-Critic																								
Zheng et al. [219]	IDDPG	MEC	Cellular	–	–	✓	–	–	–	–	Indep.	–	–	–	–	✓	–	–	–	Single	✓	Energy budget	✓	Simulation
Gong et al. [45]	IDDPG	VEC	V2X	✓	✓	✓	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	✓	Deadline	✓	Simulation
He et al. [57]	IDDPG	UAV-MEC	Air-Ground	–	–	–	–	–	✓	–	Indep.	–	✓	–	–	–	–	–	–	Single	✓	Kinematic	–	Simulation
MARL – CTDE Value Decomposition																								
Raivi & Moh [136]	VDN	UAV-MEC	Air-Ground	✓	–	–	–	–	–	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	✓	Kinematic; Energy budget	–	Simulation
Yao et al. [194]	VDN	MEC	Cellular	✓	–	✓	✓	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	–	Interference	–	Simulation
Salehi et al. [142]	VDN	MEC	Cellular	–	–	✓	–	–	–	–	Indep.	✓	–	–	–	–	✓	–	–	Multi	–	QoS	–	Simulation
Gui et al. [49]	VDN	UAV-MEC	Air-Ground	–	✓	✓	–	–	–	–	Indep.	–	✓	–	✓	✓	–	–	–	Multi	✓	Energy budget	–	Simulation
Yu et al. [200]	QMIX	UAV-MEC	Air-Ground	✓	–	–	–	–	✓	–	Indep.	✓	–	–	–	–	–	–	–	Single	✓	Kinematic; Energy budget	–	Simulation
Jiang et al. [72]	QMIX	MEC	Cellular	–	–	✓	–	–	–	–	Indep.	–	–	–	✓	–	–	–	–	Single	–	Deadline	–	Simulation
Tian et al. [161]	QMIX	UAV-MEC	Air-Ground	–	–	✓	✓	–	–	–	Indep.	–	–	–	✓	–	–	–	–	Single	✓	Energy budget	–	Simulation

Continued on next page

Table 5. (continued)

Reference	Algorithm	Scenario	Network	Decision Task							Task Model	Optimization Objectives								Obj. Type	Mobility	Constraints	Scalability	Evaluation
				Offload	*Schedule*	*Allocate*	*Cache*	*Migrate*	*Traj.*	*Other*		*Latency*	*Energy*	*Cost*	*Thput.*	*AoI*	*Res.Util.*	*Succ.R.*	*Other*					
Ning et al. [124]	QMIX	IIoT-MEC	Cellular	–	–	✓	–	–	–	–	Indep.	–	–	–	✓	✓	–	–	–	Multi	✓	–	–	Simulation
Guo et al. [50]	QMIX	VEC	V2X	✓	–	✓	–	–	–	–	Indep.	–	–	–	–	✓	–	–	–	Single	✓	Energy budget	–	Simulation
MARL – CTDE Centralized Critic																								
He et al. [59]	MADDPG	MEC	Cellular	–	–	–	✓	–	–	–	Indep.	✓	–	–	–	–	✓	–	–	Multi	–	–	–	Simulation
Peng et al. [127]	MADDPG	UAV-MEC	Air-Ground	✓	–	✓	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	✓	Deadline	–	Simulation
Hu et al. [66]	MADDPG	VEC	V2X	✓	–	✓	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	✓	Deadline; Interference	✓	Simulation
Ren et al. [140]	MADDPG	MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	✓	✓	–	–	–	✓	–	–	Multi	–	QoS	✓	Simulation
Subhan et al. [149]	COMA	MEC	Cellular	–	–	✓	✓	–	–	–	Indep.	✓	–	–	✓	–	–	–	–	Multi	✓	Bandwidth	–	Trace-driven
Liu et al. [101]	COMA	MEC	Cellular	✓	–	–	✓	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	✓	Energy budget	✓	Simulation
Hu et al. [67]	COMA	VEC	V2X	✓	✓	–	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	–	Deadline	✓	Simulation
Cui et al. [25]	COMA	VEC	V2X	✓	–	–	–	✓	–	–	Indep.	–	✓	–	–	–	–	–	–	Single	✓	Deadline	✓	Simulation
Yao et al. [193]	MAPPO	Edge-Cloud	Cellular	✓	–	✓	–	–	–	–	Indep.	✓	–	✓	–	–	–	–	–	Multi	–	–	✓	Testbed
Li et al. [94]	MAPPO	Edge-Cloud	WiFi/Cell.	–	✓	✓	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	–	–	✓	Trace-driven
Ji et al. [70]	MAPPO	MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	–	Deadline; Energy budget	–	Simulation
Kang et al. [77]	MAPPO	UAV-MEC	Air-Ground	✓	–	✓	✓	–	–	–	Indep.	–	–	–	–	–	✓	–	–	Single	–	QoS; Energy budget;	–	Simulation
Huang et al. [68]	MAPPO	Edge-Cloud	WiFi/Cell.	✓	✓	✓	–	–	–	–	Indep.	–	–	–	–	✓	–	–	–	Single	–	QoS	✓	Trace-driven
Li et al. [88]	MAPPO	MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	–	Deadline	✓	Simulation

✓ = Addressed; – = Not addressed. **Traj.** = Trajectory. **Indep.** = Independent tasks. **DAG** = DAG-structured tasks. **Thput.** = Throughput. **AoI** = Age of Information. **Res.Util.** = Resource Utilization. **Succ.R.** = Success Rate. **Single** = Single-objective. **Multi** = Multi-objective.

The choice between single-agent and multi-agent architectures depends on system scale, objective alignment, and communication infrastructure. SARL is well-suited when: (i) the number of managed entities (edge servers, IoT clusters, network slices) is moderate, allowing tractable global state representation; (ii) all system components share a coherent optimization objective expressible as a unified reward function (e.g., minimizing system-wide energy consumption, maximizing aggregate throughput); (iii) reliable communication exists between edge nodes and a central controller for policy dissemination. MARL becomes necessary when: (i) system scale grows to hundreds or thousands of entities, rendering global state spaces intractable for centralized policy networks; (ii) heterogeneous edge domains exhibit conflicting local objectives that cannot be reconciled through a single global policy (e.g., cross-operator federations where each operator maximizes its own profit, smart cities where traffic management and energy grids have fundamentally different criteria); (iii) communication latency or bandwidth constraints prevent timely global state aggregation. Independent learning suits weakly-coupled scenarios where agents have minimal mutual influence, while CTDE methods are essential when strong inter-agent dependencies demand coordinated policies despite distributed execution.

The progression from SARL to MARL reflects the growing scale and heterogeneity of IoT-Edge-Cloud systems. However, all methods discussed in this section assume free access to training data, whether through centralized collection (SARL) or global state access via centralized critics (CTDE). In practice, privacy regulations, communication constraints, and data sovereignty requirements often render such data centralization infeasible. Section 5 addresses this challenge by introducing the federated learning paradigm, which enables collaborative training without sharing raw data.

5 Federated Training Paradigm

The standard training paradigm assumes unrestricted access to training data. However, this assumption is increasingly untenable in real-world IoT-Edge-Cloud deployments due to three fundamental constraints: (i) *privacy regulations* such as General Data Protection Regulation (GDPR) and Health Insurance Portability and Accountability Act (HIPAA) prohibit raw data transmission across administrative boundaries [216]; (ii) *communication costs* make continuous data uploading from resource-constrained IoT devices economically infeasible [190]; (iii) *data sovereignty requirements* mandate that data remains locally stored within specific jurisdictions [114].

Federated learning addresses these constraints by inverting the traditional paradigm: rather than moving data to models, it moves models to data. Each node maintains local training data and periodically exchanges only model updates with an aggregator, ensuring raw data never leaves its origin while enabling collaborative learning.

Figure 3 illustrates the taxonomy under this paradigm. *Centralized Federated Architectures* (Section 5.1) employ a central aggregator for weighted aggregation, providing faster convergence at the cost of potential bottlenecks. *Decentralized Federated Architectures* (Section 5.4) eliminate central aggregation through peer-to-peer exchange, offering superior fault tolerance at the cost of slower convergence.

5.1 Centralized Federated Reinforcement Learning Architectures

Centralized federated architectures implement a hierarchical topology where a central aggregator coordinates training across distributed computing nodes [114]. This architecture is particularly prevalent in IoT-Edge-Cloud scenarios where: (i) a trusted central entity (e.g., cloud service provider, edge orchestrator) can be designated as the aggregator [195]; (ii) communication infrastructure supports reliable uplink/downlink channels between nodes and the central server [114]; (iii) the number of participating nodes is sufficiently large that the communication overhead of centralized aggregation is outweighed by faster convergence compared to decentralized alternatives [9].

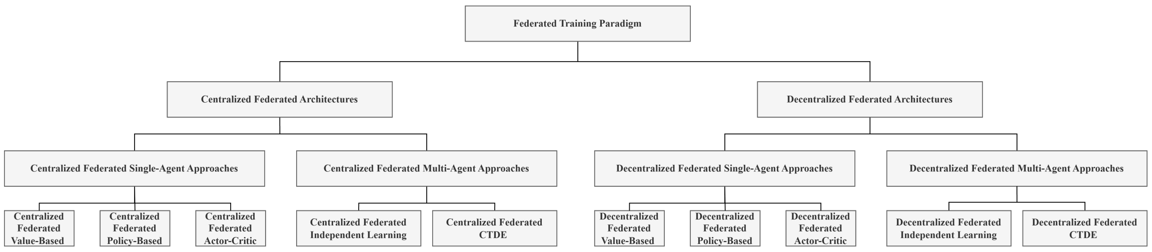
Federated Training Paradigm
Centralized Federated Architectures
Decentralized Federated Architectures
Centralized Federated Single-Agent Approaches
Centralized Federated Multi-Agent Approaches
Decentralized Federated Single-Agent Approaches
Decentralized Federated Multi-Agent Approaches
Centralized Federated Value-Based
Centralized Federated Policy-Based
Centralized Federated Actor-Critic
Centralized Federated Independent Learning
Centralized Federated CTDE
Decentralized Federated Value-Based
Decentralized Federated Policy-Based
Decentralized Federated Actor-Critic
Decentralized Federated Independent Learning
Decentralized Federated CTDE

Fig. 3. Taxonomy of DRL methods under federated training paradigm.

5.1.1 *Architecture Overview and Mechanisms.*

Central Aggregator Architecture. The centralized federated architecture employs a central aggregator (typically a cloud server or regional edge controller [182]) that coordinates distributed participants (edge servers, IoT gateways, or capable devices) through iterative training rounds. At round t, the aggregator broadcasts global model θ^t to selected clients $\mathcal{S}^t \subseteq \mathcal{N}$; each client $k \in \mathcal{S}^t$ performs local training on private dataset $\mathcal{D}_k$ for E local epochs, computing updated parameters θ_k^{t+1}; clients upload local updates $\Delta_k^t = \theta_k^{t+1} - \theta^t$ to the aggregator, which produces new global model θ^{t+1} through weighted aggregation.

Three key design parameters govern this architecture: *client participation fraction C* balances gradient estimate quality against communication overhead [114]; *local epochs E* reduces communication frequency but may cause client drift under heterogeneous data distributions [195]; *batch size B* stabilizes local training but reduces exploration. The *synchronization strategy* further differentiates designs: *synchronous aggregation* ensures consistent global models but suffers from straggler delays, while *asynchronous aggregation* incorporates updates as they arrive, improving throughput at the cost of staleness that can be partially mitigated through staleness-aware weighting [125, 146].

Centralized Aggregation Algorithms. The aggregation algorithm defines how the central server combines local updates into the global model, directly determining convergence speed and robustness to data heterogeneity.

Federated Averaging (FedAvg) [114] is the foundational aggregation algorithm. The server computes a weighted average $\theta^{t+1} = \sum_{k \in \mathcal{S}^t} \frac{n_k}{n} \theta_k^{t+1}$, where n_k is client k's local dataset size, ensuring clients with more data have proportionally greater influence. FedAvg's simplicity has made it the default baseline [195], though it may converge slowly under severe data heterogeneity [91].

Federated Proximal (FedProx) [91] addresses client drift by adding a proximal term $\frac{\mu}{2}\|\theta - \theta^t\|^2$ to each client's local objective, where $\mu > 0$ penalizes deviation from the global model. This regularization limits local model drift, improving convergence robustness under heterogeneous data distributions.

Adaptive Federated Optimization methods (FedAdam, FedYogi, FedAdagrad) [139] apply adaptive learning rate techniques at the server level. Rather than simple averaging, the server maintains momentum-based statistics of aggregated gradients, tracking first and second moments to dynamically adjust per-parameter learning rates. These methods accelerate convergence on heterogeneous objectives by automatically adapting to different learning rate requirements across clients [191].

Convergence and Communication Analysis. Centralized federated learning introduces unique convergence challenges absent in standard training. The key issue is *client drift*: when clients perform multiple local updates on heterogeneous data, local models diverge from the global optimum, with drift accumulating across rounds [195].

Theoretical analysis establishes convergence under standard assumptions (strongly convex loss, Lipschitz gradients, bounded variance), but convergence degrades with data heterogeneity and excessive local computation [195]. For DRL applications, non-stationarity and sample correlation further complicate convergence, requiring careful tuning of local epoch numbers to avoid catastrophic divergence from extrapolation error [153].

Communication efficiency is critical in bandwidth-constrained IoT-Edge scenarios, as each round incurs bidirectional transfer of model parameters at megabyte scale across thousands of devices [130]. Techniques including gradient compression, periodic aggregation, and client sampling reduce communication overhead but typically trade off convergence speed or final model quality, requiring careful balance based on network conditions and application requirements [130].

5.1.2 ***Centralized Federated Single-Agent Approaches.***

Federating SARL methods for IoT-Edge-Cloud resource management presents unique challenges beyond standard federated supervised learning. DRL training involves sequential decision-making where each agent interacts with an environment, collects trajectories (sequences of state-action-reward transitions), and updates its policy based on these experiences. In a federated setting, edge nodes execute local policies to manage their respective resources, generating trajectories from their local environments (e.g., local workload patterns, network conditions, device characteristics) [5]. The core challenge is enabling these distributed nodes to collaboratively learn a unified global policy without sharing raw trajectory data [191].

The federated single-agent paradigm assumes all nodes ultimately deploy the same global policy π_θ but train on heterogeneous local environments. For instance, in a multi-region edge content caching system, each edge region manages its cache using the same policy architecture, but local request patterns (state distributions) and cache hit rewards differ significantly across regions. Federated learning enables cross-region knowledge transfer while respecting data locality constraints [191].

Federated Value-Based Methods. Federating DQN-family methods applies federated aggregation to Q-networks. Each edge node k maintains a local Q-network Q_{θ_k} and experience replay buffer $\mathcal{D}_k$. During local training, nodes sample mini-batches from local buffers and perform standard DQN updates. After local training, nodes upload Q-network parameters to the central aggregator, which combines them into a global Q-network that is redistributed for the next training round.

Tong et al. [162] employed federated DQN with FedAvg-based aggregation to jointly minimize response time and energy consumption in multi-objective DAG task offloading, combining edge rank sorting with automated hyperparameter tuning for distributed collaborative learning. Further federated DQN applications address edge caching [217], dependency-aware vehicular fog offloading [117], and asynchronous offloading in space-assisted vehicular networks [125].

However, federated value-based methods face the challenge of heterogeneous state-action distributions. When edge nodes observe different state distributions, the aggregated Q-function may average conflicting value estimates [180]. For example, an action might have a high Q-value in a high-bandwidth region but a low Q-value in a low-bandwidth region. Simple aggregation produces a mediocre Q-estimate, potentially ineffective in both regions [180]. This challenge motivates personalized federated learning approaches or client clustering strategies based on environment similarity.

Federated Policy-Based Methods. Policy gradient methods (PPO) map more naturally to federated learning than value-based methods because policies can be directly averaged, and policy gradient estimators naturally aggregate across distributed rollouts. Each edge node k collects trajectories $\tau_k = \{(s_t, a_t, r_t)\}$ from its local environment using the current global policy π_θ, computes local policy gradients $\nabla_\theta J_k(\theta)$, and performs gradient ascent steps. The aggregator averages local policy parameters (or gradients) to update the global policy.

Wei et al. [179] employed federated PPO with personalized FedAvg aggregation to minimize delay and energy consumption in low-altitude vehicular fog computing through attention-based region-aware personalization. Shen et al. [146] developed asynchronous federated PPO with staleness-aware FedProx aggregation for UAV-assisted DAG task offloading.

A unique advantage of federated policy gradient methods is compatibility with importance sampling corrections [34]. Since nodes may perform multiple local updates before synchronization, the on-policy assumption (trajectories collected by the current policy) is violated when using stale global policies for local rollouts [114]. Standard off-policy corrections (e.g., importance sampling weights) can be applied during local advantage estimation to account for policy lag, maintaining convergence guarantees [34].

Federated Actor-Critic Methods. Off-policy actor-critic methods (DDPG, TD3, SAC) present both opportunities and challenges for federation. The opportunity lies in experience replay, which allows nodes to perform many local updates on their replay buffers before communicating, amortizing communication costs. The challenge is that both actor and critic networks can be federated, and critic networks are particularly sensitive to distribution shift as different nodes observe different state-action distributions.

Li et al. [85] employed federated DDPG with FedAvg to minimize task response delay and energy consumption in UAV-MEC offloading, with further applications in fog radio access networks [209] and edge server sleep scheduling [65]. Qin et al. [133] proposed differentiated federated SAC with trend-model-based FedAvg to minimize packet delay in Space-Air-Ground Integrated Network (SAGIN), with further applications in vehicular MEC [120] and train-to-train communications [89].

The key design consideration for federated actor-critic methods is determining which components to federate [130]. While actor networks can benefit from federation (as policies often generalize across similar environments), critic networks are more environment-specific [191]. Federated critics may suffer from averaging conflicting value estimates when nodes observe substantially different state-action distributions [191]. The choice between full federation (both actor and critic), partial federation (actor only), or hybrid approaches (federated actor with personalized critics) depends on environment similarity and communication constraints.

5.1.3 ***Centralized Federated Multi-Agent Approaches***. Federating MARL introduces an additional layer of complexity. The system must not only handle federated learning across edge nodes but also coordinate multiple agents within each node's local environment [141]. This scenario arises naturally in hierarchical IoT-Edge-Cloud systems. For example, consider a smart city deployment where each city district operates an edge cluster managing traffic lights, streetlights, and surveillance cameras. Each district wants to learn coordinated policies for its infrastructure (multi-agent problem) while collaborating with other districts (federated learning) without sharing sensitive citizen data.

Federated independent MARL (e.g., Federated IPPO, Federated IDQN) is essentially equivalent to multiple parallel instances of federated SARL, as each agent independently trains on local data and aggregates via standard federated mechanisms without any inter-agent coordination. We therefore focus this section on federated CTDE, where coordination structures (mixing networks, centralized critics) must also be federated, introducing unique design challenges absent in SARL. Specifically, CTDE methods require access to global state or joint observations during training (e.g., QMIX's mixing networks, MADDPG's centralized critics). In federated settings, sharing this global information across edge nodes violates privacy constraints, yet CTDE methods fundamentally rely on centralized information during training.

The solution is hierarchical information aggregation. Within each edge node k, local agents can access local global state (observations of all agents within that node), enabling local CTDE training. The central federated aggregator combines these locally-trained CTDE structures. For instance, in Federated QMIX, each edge node k maintains local agent networks $\{Q_i^k\}$ and a local mixing network f_{mix}^k that combines them using local global state. During federation, both agent networks $\{Q_i^k\}$ and mixing networks $\{f_{mix}^k\}$ are aggregated across nodes through the federated aggregation mechanism to produce global models that are redistributed for the next training round.

Yin et al. [197] employed federated QMIX with loss-weighted FedAvg to minimize migration latency and maximize task completion rate in mobile crowdsensing, integrating graph attention networks to extract DAG task dependencies while aggregating parameters based on loss function ratios. A further federated QMIX application addresses model-aided multi-UAV trajectory planning in IoT networks [16]. Lei et al. [84] developed federated MADDPG with loss-weighted FedAvg to maximize task completion rate in vehicular edge computing, proving the task offloading problem forms a non-cooperative potential game to ensure Nash equilibrium convergence. Additional federated MADDPG applications include joint UAV 3-D trajectory and resource allocation in marine IoT networks [160] and communication resource allocation in vehicular networks [104]. Zeng et al. [204] proposed federated MAPPO with loss-weighted FedAvg to minimize average offloading delay in Reconfigurable Intelligent Surface (RIS)-assisted V2X networks, jointly optimizing RIS reflection coefficients and beamforming vectors. Further federated MAPPO works address adaptive social metaverse streaming [107] and radio resource management in industrial 6G subnetworks [112].

A critical design question is whether to federate the centralized training components (mixing networks, centralized critics) [111]. Current practice suggests federating both decentralized actors and centralized coordinators, as coordination patterns (e.g., how to balance load among agents) often generalize across environments even when state distributions differ [141]. However, this remains an active research area with limited theoretical guidance.

5.2 Detailed Comparative Analysis

Centralized federated architectures enable privacy-preserving collaborative DRL training through a trusted aggregator that coordinates model updates across distributed edge nodes. Their effectiveness for IoT-Edge-Cloud resource management varies across the four system-level dimensions analyzed below.

5.2.1 ***Scalability.*** Centralized federation supports moderate-scale deployments (typically fewer than several hundred nodes) but faces inherent bottlenecks as system size grows [114, 154]. The central aggregator must receive, process, and broadcast model updates from all participating clients each round, with communication cost scaling linearly with client count [91]. Partial client participation (selecting a fraction C of clients per round) alleviates bandwidth pressure but introduces gradient noise that may slow convergence [114]. Among DRL method categories, value-based methods (Federated DQN) produce compact parameter updates suitable for bandwidth-constrained aggregation [162, 217], while actor-critic methods (Federated DDPG, SAC) double the communication payload by federating both actor and critic networks [133]. Federated CTDE methods (Federated QMIX, MADDPG, MAPPO) add further scalability pressure by federating mixing networks or centralized critics alongside agent policies [84, 197, 204].

5.2.2 ***Convergence.*** *Speed.* Synchronized global aggregation enables faster convergence than decentralized alternatives by ensuring all clients operate from the same global model each round [91, 114]. However, convergence speed varies across method categories: Federated DQN faces heterogeneous Q-value distributions across clients that may conflict during averaging, slowing convergence when edge environments differ significantly [117, 162]; Federated PPO benefits from importance sampling corrections that handle policy lag from stale global policies [146, 179]; Federated actor-critic methods achieve faster convergence through off-policy replay that tolerates the delay between global model broadcasts, with SAC's entropy regularization further improving convergence robustness [89, 133]. Adaptive aggregation methods (FedAdam, FedYogi) accelerate convergence through momentum-based statistics, particularly effective for actor-critic methods where gradient magnitudes vary substantially between actor and critic [139]. *Stability.* Theoretical convergence guarantees exist under standard assumptions (bounded heterogeneity, Lipschitz continuity), though their applicability to DRL requires empirical validation due to non-stationarity and correlated sampling [91]. Client drift from excessive local epochs E degrades stability, particularly severe for value-based methods where local Q-networks diverge rapidly under heterogeneous replay distributions [217]. FedProx mitigates drift through proximal regularization, improving stability for heterogeneous edge deployments [91, 125]. For federated MARL, convergence stability depends on whether centralized training components (mixing networks in QMIX, centralized critics in MADDPG/MAPPO) are federated or kept local, with current practice federating both but limited theoretical understanding of the implications [84, 197].

5.2.3 *Overhead.*

Communication overhead dominates in centralized federation: each round requires uploading local model updates and downloading the global model, with round frequency directly trading off against convergence quality [114]. Increasing local epochs E reduces communication rounds but risks client drift; value-based methods require careful tuning of E to avoid excessive Q-value divergence, while policy-based methods should align E with episode lengths [162, 179]. Infrastructure overhead is moderate: a single aggregation server (cloud or regional controller) with sufficient bandwidth and computation for weighted averaging [91]. Privacy mechanisms add varying overhead: differential privacy introduces noise with minimal computation [179]; secure aggregation requires cryptographic computation scaling with client count [11]; homomorphic encryption incurs orders-of-magnitude computational overhead but protects against malicious aggregators [92, 186]. Federated CTDE methods compound overhead by requiring aggregation of multiple model components (agent networks + mixing/critic networks) per round [197, 204].

5.2.4 *Adaptability.*

Centralized federation enables coordinated adaptation through global model updates that propagate learned behaviors across all clients [114]. However, a single global model may struggle to adapt to heterogeneous local conditions when edge regions exhibit drastically different resource characteristics or workload patterns [91]. Personalized federated learning approaches (per-client fine-tuning, clustering-based aggregation) address this by allowing local specialization while retaining shared representations [14]. Off-policy methods (Federated DDPG, SAC) exhibit stronger adaptability than on-policy methods (Federated PPO) because local replay buffers retain historical experience that remains useful despite global model updates [133]. Asynchronous aggregation variants improve adaptability to heterogeneous client speeds but introduce staleness that may degrade adaptation quality for fast-changing environments [125, 146]. Federated CTDE methods benefit from coordinated multi-agent adaptation through federated centralized components but risk propagating locally optimal coordination patterns that conflict across heterogeneous client environments [84, 204].

5.3 Design Guidelines.

Aggregation algorithm selection should match environment heterogeneity: standard FedAvg for homogeneous edge deployments [114]; FedProx for heterogeneous data distributions with proximal coefficients tuned to distribution divergence [91, 125]; adaptive methods (FedAdam, FedYogi) when convergence speed is critical and gradient statistics vary across clients [139]. *Communication-convergence trade-off* requires tuning local epochs E to balance round frequency against client drift, with value-based methods typically requiring smaller E than actor-critic methods due to faster Q-value divergence [162]. *Federation scope* for actor-critic methods should be determined by environment similarity: federate actor only when environments differ substantially (preserving local critics), full federation (actor + critic) when environments are reasonably aligned [85, 133]. *Privacy mechanism selection* should trade off guarantee strength against overhead: differential privacy for resource-constrained deployments, secure aggregation for moderate overhead with strong guarantees, and homomorphic encryption for maximum protection in sensitive applications [92, 179].

5.4 Decentralized Federated Reinforcement Learning Architectures

Decentralized federated architectures implement peer-to-peer model aggregation where edge nodes exchange model updates directly with neighbors [182]. This architecture is particularly prevalent in IoT-Edge-Cloud scenarios where: (i) no mutually trusted central entity exists across all participating nodes (e.g., cross-operator edge federations, collaborations between competing service providers) [9]; (ii) communication infrastructure supports reliable peer-to-peer links but edge-to-cloud uplinks are bandwidth- constrained or high-latency [182]; (iii) the number of participating nodes is sufficiently large that the scalability and fault tolerance advantages of decentralized topologies outweigh the slower convergence disadvantage [9].

5.4.1 *Architecture Overview and Mechanisms*.

Peer-to-Peer Aggregation Architecture. The decentralized federated architecture comprises distributed participants (edge servers, gateways, or capable IoT devices [166]) each maintaining local data and model replicas, connected through a communication topology that can be static or dynamically adjusted [182]. Training proceeds in locally-defined rounds without global synchronization: node i performs E local epochs on private dataset $\mathcal{D}_i$, then selects neighbors from its neighborhood set $\mathcal{N}_i$ and combines parameters through pairwise averaging $\theta_i \leftarrow (\theta_i + \theta_j)/2$ to achieve local consensus, typically executing asynchronously to tolerate network latency and heterogeneous processing speeds.

Three key design parameters govern this architecture: *topology degree k* trades convergence speed against communication overhead [9]; *local epoch count E* balances learning efficiency with parameter drift; *exchange frequency* impacts consensus speed versus bandwidth consumption [56]. The synchronization strategy further differentiates designs: *synchronous exchange* ensures consistent parameter combination but introduces straggler sensitivity, while *asynchronous exchange* tolerates heterogeneous speeds and intermittent connectivity at the cost of staleness from neighbors at different training stages.

Distributed Aggregation Mechanisms. Distributed aggregation mechanisms define how nodes combine parameters from neighbors to achieve global consensus, directly determining convergence speed, robustness to data heterogeneity, and adaptability to network dynamics.

Gossip averaging [12] is the foundational decentralized aggregation mechanism. In each communication round, node i selects a neighbor j from $\mathcal{N}_i$, and both nodes perform pairwise averaging: $\theta_i \leftarrow (\theta_i + \theta_j)/2$. This update rule ensures network-wide parameter average remains invariant, with all nodes converging exponentially to this average under connected graph assumptions. Gossip averaging requires no global coordination, supports asynchronous execution, and naturally tolerates node failures. However, it may converge slowly under severe data heterogeneity [9].

Consensus-based methods [9] achieve parameter aggregation through distributed protocols providing stronger security guarantees. Blockchain-inspired mechanisms ensure update integrity through distributed ledgers, achieving Byzantine fault tolerance [135]. Optimization frameworks such as Alternating Direction Method of Multipliers

(ADMM) provide alternative approaches through constrained optimization, converging faster than gossip for convex objectives [182]. These methods offer stronger theoretical guarantees but increase computational complexity through verification overhead.

Gradient tracking [134] addresses the limitation that parameter averaging may fail to minimize global objectives when local objectives differ significantly. Each node maintains both parameter and gradient estimates, with the protocol ensuring convergence to the true global gradient despite computing only local gradients. This enables gradient descent on the global objective while observing only local data, achieving linear convergence for strongly convex objectives [87].

Convergence and Communication Analysis. Unlike centralized federated learning, where global aggregation ensures parameter consistency, decentralized architectures face consensus error from multi-hop information propagation [80]. When nodes perform local updates on heterogeneous data between peer exchanges, parameters across the network converge only asymptotically, with the spectral gap of the communication graph determining the exponential decay rate of consensus error [9]. Data heterogeneity and excessive local computation exacerbate this challenge. For DRL, the peer-to-peer nature introduces additional complications: nodes average parameters from neighbors whose policies have evolved independently on different environment interactions, creating misaligned value estimates and policy distributions [191].

Communication efficiency in decentralized settings exhibits distinct characteristics from centralized approaches. Topology design determines communication patterns, with node degree k controlling per-round volume and graph diameter determining convergence speed, while asynchronous exchange and network partition handling introduce unique trade-offs [12]. However, low-latency inter-edge links enable peer exchanges orders of magnitude faster than edge-to-cloud round trips, partially offsetting the increased number of communication rounds for time-sensitive applications [9].

5.4.2 ***Decentralized Federated Single-Agent Approaches.***

Decentralized federated SARL extends the single global policy paradigm to peer-to-peer topologies, enabling edge nodes to collaboratively learn unified resource management policies without central coordination. Each edge node k maintains a local DRL agent (Q-network Q_{θ_k}, policy π_{θ_k}, or actor-critic pair), training on its private environment. Unlike centralized federation, nodes exchange parameters directly with neighbors through gossip protocols or structured topologies, gradually reaching consensus through pairwise averaging: $\theta_k \leftarrow (\theta_k + \theta_j)/2$.

Chai et al. [14] employed decentralized federated DQN with gossip-based attention-weighted aggregation to minimize delay and energy in vehicle-assisted edge computing, with a further application in privacy-preserving clinical decision systems [187]. Jeong et al. [69] employed consensus-based decentralized federated PPO with committee voting for Byzantine-resilient 5G resource scaling. Zhou et al. [220] proposed decentralized federated DDPG with gossip-based weighted selective aggregation for edge caching, with a further application in user-centric dynamic MEC control [198].

The key design consideration for decentralized federated SARL is algorithm category selection and model component federation strategy. Off-policy methods (DQN, SAC) provide significant advantages as experience replay enables nodes to continue learning from local buffers while waiting for gossip exchanges, tolerating asynchronous communication and stale neighbor parameters [8]. On-policy methods (PPO) face more severe convergence issues as different node policies violate on-policy assumptions [8]. For actor-critic methods, actor networks can benefit from decentralized federation as policies generalize across similar environments, while federated critics are susceptible to averaging conflicting value estimates across heterogeneous state-action distributions [191]. For highly heterogeneous edge environments, gradient tracking provides an alternative through exchanging gradient estimates rather than parameters, optimizing the true global objective at the cost of increased computational overhead [191].

5.4.3 *Decentralized Federated Multi-Agent Approaches.*

Similar to centralized federated settings (Section 5.1.3), decentralized federated independent MARL is essentially equivalent to decentralized federated SARL, as each agent independently trains and aggregates. Decentralized federated CTDE represents the most complex intersection in our taxonomy, requiring simultaneous coordination across three dimensions: multi-agent coordination via centralized training components (mixing networks, centralized critics), federated learning under privacy constraints, and decentralized aggregation without central coordination. To our knowledge, existing federated CTDE for IoT-Edge-Cloud resource management exclusively adopts centralized aggregation; fully decentralized federated CTDE remains largely unexplored, making this an open research frontier.

Several fundamental barriers impede progress. CTDE methods fundamentally rely on centralized components during training, while decentralized peer-to-peer architectures lack such central entities, creating an inherent architectural incompatibility [9, 108]. Convergence analysis must simultaneously account for multi-agent non-stationarity, consensus dynamics of decentralized topologies, and heterogeneity across nodes [9, 108]. Distributed credit assignment must operate under potentially stale policies using only local information [39]. Promising research directions include hierarchical decomposition strategies separating intra-node and inter-node coordination, gradient tracking extensions for multi-agent settings, communication-efficient partial information sharing protocols, and standardized benchmarks tailored to decentralized edge deployments [9].

5.5 Decentralized Federated RL: Detailed Comparative Analysis and Design Guidelines

Decentralized federated architectures eliminate the central aggregator, enabling peer-to-peer model exchange across edge nodes. Their effectiveness for IoT-Edge-Cloud resource management varies across the four system-level dimensions analyzed below.

5.5.1 *Scalability.*

Decentralized federation achieves the strongest scalability among all federated architectures. Each node exchanges parameters with only a fixed number of neighbors regardless of network size, achieving near-linear scaling to thousands of edge nodes [95]. This contrasts sharply with centralized architectures that encounter server bandwidth saturation as client count grows [114]. Topology degree k (number of neighbors per node) controls the scalability-convergence trade-off: lower k reduces per-node communication but slows consensus propagation [79]. Among DRL methods, off-policy methods (Decentralized Federated DQN, DDPG) are particularly well-suited because they tolerate the stale parameters inherent in asynchronous peer exchange [198, 220]. On-policy methods (Decentralized Federated PPO) face greater challenges as policy lag from multi-hop propagation degrades importance sampling corrections [69]. Decentralized Federated MARL with CTDE remains an open research frontier due to the fundamental architectural incompatibility between CTDE's centralized training requirements and decentralized federation's lack of global coordination [171].

5.5.2 *Convergence.*

Speed. Multi-hop consensus propagation requires more communication rounds to reach equivalent policy quality compared to centralized approaches, as information must diffuse through the topology rather than being instantly broadcast [79, 95]. Convergence speed depends critically on topology design: well-connected topologies (higher degree, lower diameter) accelerate consensus but increase per-node communication overhead [79]. Gossip averaging achieves exponential convergence for convex objectives [14], but DRL's non-convex, non-stationary landscape creates additional challenges. Among methods, Decentralized Federated DQN with attention-weighted gossip aggregation selectively emphasizes high-performing neighbors to accelerate convergence [14]; Decentralized Federated PPO with committee voting achieves Byzantine-resilient convergence for 5G resource management [69]; Decentralized Federated DDPG with selective weighted aggregation focuses parameter exchange on relevant peers for faster convergence in edge caching [198]. *Stability.* Limited theoretical understanding for decentralized DRL settings creates uncertainty: convergence analysis faces open questions due to policy non-stationarity, temporal correlation in DRL data, and interaction between consensus error and learning dynamics [95]. Consensus-based aggregation with blockchain-inspired verification provides Byzantine fault tolerance but with higher computational complexity [187]. Gradient tracking methods converge to the true global gradient despite local-only computation, offering the strongest theoretical stability for highly heterogeneous environments at the cost of increased per-iteration computation [187].

5.5.3 ***Overhead.*** Decentralized federation distributes communication cost across the topology, eliminating centralized bottlenecks but introducing different overhead patterns [95]. Per-node communication scales with topology degree k rather than total network size, enabling efficient operation in bandwidth-constrained edge mesh networks [79]. Infrastructure overhead is minimal: no dedicated aggregation server, leveraging existing inter-edge communication channels [14, 198]. However, implementation complexity is higher than centralized architectures: distributed consensus protocols, sophisticated failure detection, and complex recovery mechanisms add development and maintenance overhead [187]. Off-policy methods (Decentralized Federated DQN, DDPG) achieve lower effective communication overhead because they tolerate asynchronous exchange and stale parameters, allowing less frequent peer communication without degrading learning quality [198, 220]. On-policy methods (Decentralized Federated PPO) require more frequent exchange to maintain policy freshness, increasing communication overhead relative to off-policy alternatives [69]. Privacy risk is distributed (no single entity observes all updates), but multi-hop parameter propagation introduces new attack surfaces requiring peer-level verification [187].

5.5.4 ***Adaptability.*** Decentralized federation excels at graceful degradation and local adaptation. Peer-to-peer topologies tolerate node failures without system-wide impact, maintaining consensus within connected components during network partitions [95, 187]. This is particularly valuable for IoT-Edge-Cloud deployments spanning unreliable infrastructure (satellite networks, rural cellular, mobile edge) [14]. Dynamic topologies that reconfigure peer connections based on environment similarity or performance metrics enable adaptive collaboration, allowing nodes in similar edge environments to cluster for more effective knowledge sharing [198]. However, the lack of global coordination prevents rapid system-wide adaptation: information about sudden global changes (e.g., major infrastructure failures, coordinated attacks) propagates slowly through multi-hop gossip [79]. Off-policy methods provide stronger local adaptability through replay buffers that retain useful historical experience during peer model updates [198, 220]. The unexplored decentralized federated CTDE frontier represents a critical gap: current methods cannot achieve coordinated multi-agent adaptation without global coordination, limiting applicability to multi-agent IoT-Edge-Cloud scenarios requiring both privacy preservation and tight inter-agent coordination.

5.6 Design Guidelines.

Topology selection should match deployment characteristics: static topologies for stable infrastructure with predictable inter-node connectivity; dynamic topologies for environments with frequent node changes, mobility, or heterogeneous link quality [79, 198]. *Aggregation mechanism selection* depends on environment heterogeneity and trust requirements: gossip averaging for reasonably aligned objectives with low overhead [14]; consensus-based methods for Byzantine fault tolerance in adversarial or multi-stakeholder deployments [187]; gradient tracking for highly heterogeneous environments despite increased computational overhead [187]. *Method selection* should favor off-policy methods (Decentralized Federated DQN, DDPG) for their natural tolerance of asynchronous communication and stale parameters [198, 220]; on-policy methods (Decentralized Federated PPO) should be reserved for scenarios where their specific advantages (clipping stability, Byzantine resilience) outweigh the communication overhead [69]. *Topology degree k* should balance convergence speed against per-node communication budget: higher k for latency-sensitive applications, lower k for bandwidth-constrained edge

5.7 Centralized vs Decentralized Federated DRL Architectures

Table 6 extends the literature comparison to works adopting the federated training paradigm. In addition to the dimensions reported in Table 5, we further record the aggregation strategy, distinguishing centralized server-based aggregation (e.g., FedAvg, FedProx) from decentralized peer-to-peer aggregation (e.g., gossip-based, consensus-based), and whether asynchronous aggregation is employed.

Table 6. Literature-Level Comparison of DRL Methods for Resource Management in IoT-Edge-Cloud Continuum (Federated Training Paradigm)

Reference	Method	Aggregation	Async	Scenario	Network	Decision Task							Task Model	Optimization Objectives								Obj. Type	Mobility	Constraints	Scalability	Evaluation
						Offload	*Schedule*	*Allocate*	*Cache*	*Migrate*	*Traj.*	*Other*		*Latency*	*Energy*	*Cost*	*Thput.*	*AoI*	*Res.Util.*	*Succ.R.*	*Other*					
Centralized Federated SARL – Value-Based																										
Tong et al. [162]	Cent-Fed-DQN	FedAvg	–	MEC	Cellular	✓	✓	–	–	–	–	–	DAG	✓	✓	–	–	–	–	–	–	Multi	–	–	✓	Trace-driven
Zhao et al. [217]	Cent-Fed-DQN	FedAvg	–	MEC	Cellular	–	–	–	✓	–	–	–	Indep.	–	–	✓	–	–	–	–	–	Single	–	–	✓	Simulation
Mishra et al. [117]	Cent-Fed-DQN	FedAvg	–	VEC	V2X	✓	✓	✓	–	–	–	–	DAG	✓	✓	–	–	–	✓	–	–	Multi	✓	Deadline	✓	Trace-driven
Pan et al. [125]	Cent-Fed-DQN	FedAvg	✓	VEC	V2X	✓	–	✓	–	–	–	–	Indep.	–	–	–	✓	–	–	–	–	Single	✓	Deadline	–	Simulation
Centralized Federated SARL – Policy-Based																										
Wei et al. [179]	Cent-Fed-PPO	FedAvg	–	VEC	V2X	✓	–	–	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	✓	Deadline	✓	Simulation
Shen et al. [146]	Cent-Fed-PPO	FedProx	✓	UAV-MEC	V2X	✓	–	–	–	–	–	–	DAG	✓	✓	–	–	–	–	–	–	Multi	✓	Deadline; Energy budget	✓	Trace-driven
Centralized Federated SARL – Actor-Critic																										
Li et al. [85]	Cent-Fed-DDPG	FedAvg	–	UAV-MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	–	Deadline; Energy budget	✓	Simulation
Zhang et al. [209]	Cent-Fed-DDPG	FedAvg	–	MEC	Cellular	✓	–	✓	–	–	–	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	–	–	✓	Simulation
Hou et al. [65]	Cent-Fed-DDPG	FedAvg	✓	VEC	V2X	–	–	–	–	–	–	✓	Indep.	✓	✓	✓	–	–	–	–	–	Multi	✓	–	✓	Trace-driven
Qin et al. [133]	Cent-Fed-SAC	FedAvg	–	MEC	Satellite	✓	–	–	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	✓	–	✓	Simulation
Moghaddasi et al. [120]	Cent-Fed-SAC	FedAvg	–	VEC	V2X	✓	–	✓	–	–	–	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	✓	Deadline; Energy budget	–	Simulation
Li et al. [89]	Cent-Fed-SAC	FedAvg	–	IIoT-MEC	V2X	–	–	✓	–	–	–	✓	Indep.	–	✓	–	✓	–	–	–	–	Multi	✓	Interference; Deadline	✓	Simulation
Centralized Federated MARL – CTDE																										
Yin et al. [197]	Cent-Fed-QMIX	FedAvg	✓	MEC	Cellular	–	✓	–	–	✓	–	–	Indep.	–	–	✓	–	–	–	✓	–	Multi	✓	Deadline	✓	Trace-driven
Chen et al. [16]	Cent-Fed-QMIX	FedAvg	–	UAV-MEC	Air-Ground	–	–	–	–	–	✓	–	Indep.	–	–	–	✓	–	–	–	–	Single	✓	Energy budget	–	Simulation
Lei et al. [84]	Cent-Fed-MADDPG	FedAvg	–	VEC	V2X	✓	–	✓	–	–	–	–	Indep.	–	–	–	–	–	–	✓	–	Single	✓	–	✓	Trace-driven
Tesfaw et al. [160]	Cent-Fed-MADDPG	FedAvg	–	UAV-MEC	Air-Ground	–	–	✓	–	–	✓	–	Indep.	–	–	–	✓	–	✓	–	✓	Multi	✓	Bandwidth	✓	Simulation
Liu et al. [104]	Cent-Fed-MADDPG	FedAvg	✓	VEC	V2X	–	–	✓	–	–	–	–	Indep.	–	–	–	✓	–	–	✓	–	Multi	✓	–	✓	Simulation
Zeng et al. [204]	Cent-Fed-MAPPO	FedAvg	–	VEC	V2X	✓	–	✓	–	–	–	–	Indep.	✓	–	–	–	–	–	–	–	Single	✓	Reliability	✓	Simulation
Long et al. [107]	Cent-Fed-MAPPO	FedAvg	–	MEC	WiFi	–	–	✓	–	–	–	–	Indep.	✓	–	–	✓	–	–	–	–	Multi	–	–	–	Testbed
Madsen et al. [112]	Cent-Fed-MAPPO	FedAvg	–	IIoT-MEC	WiFi/Cell.	–	–	✓	–	–	–	–	Indep.	–	–	–	✓	–	–	–	–	Single	✓	QoS	✓	Simulation
Decentralized Federated SARL – Value-Based																										
Chai et al. [14]	Dec-Fed-DQN	Gossip	–	UAV-MEC	Cellular	✓	✓	✓	–	–	✓	–	Indep.	✓	✓	–	–	–	–	–	–	Multi	✓	Kinematic	–	Simulation
Xue et al. [187]	Dec-Fed-DQN	Consensus	–	MEC	WiFi/Cell.	–	–	✓	–	–	–	–	Indep.	–	–	–	–	–	–	–	–	Single	–	Energy budget; Deadline; Privacy	–	Trace-driven
Decentralized Federated SARL – Policy-Based																										
Jeong et al. [69]	Dec-Fed-PPO	Consensus	–	MEC	Cellular	–	–	✓	–	–	–	–	Indep.	–	–	–	✓	–	–	–	–	Multi	–	QoS	–	Simulation
Decentralized Federated SARL – Actor-Critic																										
Zhou et al. [220]	Dec-Fed-DDPG	Gossip	–	MEC	Cellular	–	✓	✓	✓	–	–	–	Indep.	✓	–	–	✓	–	–	–	–	Multi	–	Bandwidth	–	Trace-driven
Yin et al. [198]	Dec-Fed-DDPG	Gossip	–	MEC	Cellular	–	✓	✓	–	–	–	–	Indep.	–	–	–	✓	–	–	–	–	Single	✓	Reliability; Interference	–	Simulation

✓ = Addressed; – = Not addressed. **Async** = Asynchronous aggregation. **Traj.** = Trajectory. **Indep.** = Independent tasks. **DAG** = DAG-structured tasks. **Thput.** = Throughput. **AoI** = Age of Information. **Res.Util.** = Resource Utilization. **Succ.R.** = Success Rate. **Single** = Single-objective. **Multi** = Multi-objective.

Table 7 provides a systematic comparison of federated DRL methods across scalability, convergence, overhead, and adaptability. The choice between centralized and decentralized federated DRL architectures depends on deployment scale, trust models, and infrastructure characteristics.

Table 7. Systematic Comparison of Federated DRL Methods for IoT-Edge-Cloud Continuum Resource Management

Architecture	Control Scope	Method Category	Representative Methods	Literature	Scalability	Convergence	Overhead	Adaptability
Centralized	SARL	Value-Based	Cent-Fed-DQN	[117, 125, 162, 217]	**Moderate.** Supports up to several hundred nodes; central aggregator bandwidth scales linearly with client count, creating bottleneck at large scale. Partial participation alleviates pressure but introduces gradient noise. Actor-critic and CTDE methods amplify payload through multi-component federation	**Good.** *Speed:* Synchronized global aggregation ensures fast convergence; off-policy methods (Fed-DDPG, Fed-SAC) tolerate aggregation delay better than on-policy (Fed-PPO); adaptive aggregation (FedAdam, FedYogi) accelerates through momentum. *Stability:* Theoretical guarantees under bounded heterogeneity; client drift from excessive local epochs degrades stability; FedProx mitigates via proximal regularization; Fed-CTDE stability under federation remains theoretically underexplored	**Moderate–High.** Per-round upload/download of model parameters; communication frequency trades off against convergence quality. Privacy mechanisms add varying overhead: differential privacy (minimal), secure aggregation (moderate), homomorphic encryption (orders of magnitude). Fed-CTDE compounds overhead by aggregating multiple model components per round	**Moderate.** Global model propagates learned behaviors across all clients; single global model may struggle with heterogeneous local conditions. Personalized FL (fine-tuning, clustering) enables local specialization. Off-policy methods (Fed-DDPG, Fed-SAC) adapt better than on-policy via local replay retention. Asynchronous variants improve heterogeneous-speed adaptation at staleness cost
		Policy-Based	Cent-Fed-PPO	[146, 179]				
		Actor-Critic	Cent-Fed-DDPG Cent-Fed-SAC	[65, 85, 209] [89, 120, 133]				
	MARL	Independent	*Equivalent to Centralized Federated SARL (cf. Section 5.1.3)*					
		CTDE	Cent-Fed-QMIX Cent-Fed-MADDPG Cent-Fed-MAPPO	[16, 197] [84, 104, 160] [107, 112, 204]				
Decentralized	SARL	Value-Based	Decent-Fed-DQN	[14, 187]	**High.** Near-linear scaling to thousands of nodes; per-node communication scales with topology degree k not network size. Off-policy methods tolerate asynchronous exchange naturally; on-policy methods face policy lag challenges. Decentralized Fed-CTDE remains an open frontier	**Moderate.** *Speed:* Multi-hop consensus requires more rounds than centralized broadcast; topology connectivity critically determines speed; gossip achieves exponential convergence for convex objectives but DRL non-convexity adds challenges. *Stability:* Limited theoretical understanding for DRL settings; consensus-based methods provide Byzantine tolerance; gradient tracking converges to true global gradient under heterogeneity	**Low–Moderate.** No centralized infrastructure; per-node communication bounded by k neighbors. Off-policy methods tolerate infrequent exchange, reducing communication. Implementation complexity is higher (consensus protocols, failure detection). Privacy risk distributed but multi-hop introduces new attack surfaces	**High.** Graceful degradation under node failures; dynamic topologies enable adaptive peer clustering by environment similarity. Slow global-change propagation through multi-hop gossip. Off-policy replay enables strong local adaptation. Decentralized Fed-CTDE gap limits coordinated multi-agent adaptation
		Policy-Based	Decent-Fed-PPO	[69]				
		Actor-Critic	Decent-Fed-DDPG	[198, 220]				
	MARL	*Independent: Equivalent to Decentralized Federated SARL (cf. Section 5.4.3). CTDE: Open research frontier.*						

Centralized federated DRL is well-suited when: (i) a mutually trusted central entity (cloud service provider, edge orchestrator) can be designated as the aggregator; (ii) deployment scale is moderate (typically fewer than several hundred nodes), where server bandwidth suffices to aggregate updates and broadcast global models; (iii) reliable high-bandwidth uplink/downlink channels exist between edge nodes and the central server; (iv) faster convergence is prioritized over fault tolerance, as centralized synchronized aggregation typically requires fewer communication rounds than decentralized consensus.

Decentralized federated DRL is preferable when: (i) no mutually trusted central entity exists (cross-operator edge federations, competing service providers); (ii) deployment scale reaches thousands of nodes, where centralized server bandwidth becomes a bottleneck; (iii) communication infrastructure supports reliable peer-to-peer links but edge-to-cloud uplinks are bandwidth-constrained or high-latency (satellite networks, rural cellular); (iv) mission-critical applications demand continued operation despite infrastructure failures, favoring decentralized architectures' graceful degradation and tolerance to network partitions.

The architectures and training paradigms discussed thus far establish the foundational design space for DRL-based resource management in the IoT-Edge-Cloud continuum. However, deploying these methods in production environments reveals orthogonal challenges that cut across architectural choices, requiring additional techniques and mechanisms to enhance the core architectures. Section 6 examines these enhancements and their applications to IoT-Edge-Cloud resource management.

6 Advanced Techniques and Enhancements

Beyond control architecture and training paradigm choices, DRL research for IoT-Edge-Cloud resource management encompasses multiple orthogonal enhancement dimensions. This section systematically reviews these advanced techniques and their applications.

6.1 Network Architectures

Standard DRL implementations employ fully-connected Multi-layer Perceptrons (MLPs) that treat timesteps independently, discarding the temporal structure inherent to resource management. Edge workloads exhibit strong temporal correlations, and resource states evolve through autoregressive dynamics. Alternative network architectures explicitly model these dependencies, improving sample efficiency and policy quality.

6.1.1 ***Recurrent Neural Networks.*** Recurrent Neural Networks (RNNs) maintain hidden states capturing historical information, with Long Short-Term Memory (LSTM) networks addressing vanishing gradients through gating mechanisms and Gated Recurrent Units (GRUs) simplifying the architecture with fewer parameters. Representative works include [2, 36, 218]. RNNs suit tasks with temporal dependencies and partial observability but can introduce Backpropagation Through Time (BPTT) complexity and susceptibility to overfitting on temporal patterns [42, 46].

6.1.2 ***Transformers and Attention Mechanisms.*** Transformers process sequences in parallel through multi-head self-attention, capturing long-range dependencies while avoiding vanishing gradients. Representative works include [14, 59, 175, 179, 197]. Transformers are particularly effective for modeling interdependencies among heterogeneous system entities (e.g., tasks, servers, and network links) but introduce quadratic attention complexity challenging edge deployment despite efficient variants [159].

6.1.3 ***Graph Neural Networks.*** Graph Neural Networks (GNNs) aggregate neighbor information over graph structures through message passing, with Graph Convolutional Networks (GCNs) and Graph Attention Networks (GATs) enabling policies to exploit network topology. Representative works include [106, 168, 181]. GNNs suit networked infrastructure with explicit topology, enabling inductive generalization to unseen graph structures, but require graph-structured observations and face computational costs scaling with graph size [1].

6.2 Training Enhancements

Conventional DRL methods typically require agents to learn policies from scratch for each new environment. Edge deployments span diverse environments, and the ability to rapidly adapt learned knowledge to new settings is critical. Training enhancements enable DRL agents to leverage prior experience, reducing sample complexity and improving generalization.

6.2.1 ***Meta-Learning.*** Meta-learning, or learning to learn, trains models that quickly adapt to new tasks with minimal data by optimizing for rapid adaptation across task distributions. For DRL, this enables learning policies that quickly specialize to new edge environments through minimal fine-tuning. Representative works include [26, 131, 145]. Meta-learning suits scenarios requiring rapid adaptation to new deployments or periodic variations but requires diverse training task distributions and incurs higher meta-training computational costs [38, 44].

6.2.2 ***Transfer Learning.*** Transfer learning reuses knowledge from source domains through pre-training policies or value functions in resource-rich environments, then fine-tuning in target edge environments, with domain adaptation techniques handling distribution shifts through adversarial training or importance weighting. Representative works include [123, 129]. Transfer learning reduces target domain sample requirements when source and target share underlying structure but may cause negative transfer when domains differ significantly [221].

6.2.3 ***Multi-Task Learning***.

Multi-task learning trains single models simultaneously on multiple related tasks through shared representations, exploiting task commonalities via soft parameter sharing, hard parameter sharing, or attention mechanisms. Representative works include [26, 201]. Multi-task learning improves sample efficiency when tasks share structure and reduces total parameters, but negative transfer may occur when tasks conflict or require different features [18].

6.3 Model Compression

DRL policies employ deep networks with millions of parameters, creating prohibitive computational and memory requirements for resource-constrained edge devices. Model compression techniques reduce model size and inference cost while preserving policy quality, enabling deployment at the lowest tiers of the edge continuum.

6.3.1 ***Knowledge Distillation***.

Knowledge distillation transfers knowledge from large teacher models to compact students by minimizing divergence between outputs, exploiting the fact that teacher soft targets contain richer information than hard targets for efficient student learning. Representative works include [21, 205]. Distillation enables cloud training with edge deployment and reduces federated learning communication overhead but introduces additional training stages and bounds student performance by teacher quality [173].

6.3.2 ***Quantization***.

Quantization reduces model size by lowering numerical precision from 32-bit floating point to 8-bit or lower integers, with post-training quantization applied after training and quantization-aware training simulating quantization effects during learning. Representative works include [206, 215]. Quantization provides memory reduction and inference speedup, particularly effective with hardware accelerators, but introduces numerical errors that may degrade performance for precision-sensitive tasks [115].

6.3.3 ***Pruning***.

Pruning removes redundant parameters through unstructured pruning of individual weights or structured pruning of entire neurons and layers, with magnitude-based selection and iterative refinement recovering performance. Representative works include [71, 122]. Pruning achieves high compression when models contain redundancy, with structured pruning providing practical speedup without specialized sparse libraries, but determining pruning rates requires experimentation and may fail for small models with limited redundancy [115].

6.4 Safety and Privacy Mechanisms

DRL policies deployed in production edge systems commonly need to satisfy hard constraints while protecting sensitive data. Safety mechanisms ensure constraint satisfaction during learning and deployment, preventing violations that may lead to service-level agreement breaches, hardware damage, or safety hazards. Privacy mechanisms safeguard training data and model information from unauthorized access and inference attacks, essential for edge systems processing sensitive data across administrative boundaries.

6.4.1 ***Constrained Reinforcement Learning***.

Constrained RL formalizes constraint satisfaction as Constrained Markov Decision Processes (CMDPs) where agents optimize cumulative reward while satisfying cumulative constraints. Representative works include [54, 184]. Constrained RL provides principled frameworks for constraint optimization in safety-critical applications and service level agreements but assumes known constraint cost functions and may struggle with multiple competing constraints [43].

6.4.2 ***Safe Exploration***.

Safe exploration ensures agents do not violate constraints during learning by monitoring policies and intervening to prevent unsafe actions through pre-computed safety constraints or conservative action prediction. Representative works include [110, 213]. Safe exploration provides runtime protection for online learning in production systems but may interfere with policy improvement through frequent intervention or prove computationally infeasible in large state spaces [43].

6.4.3 ***Homomorphic Encryption***.

Homomorphic Encryption (HE) enables computation directly on encrypted data without decryption, allowing aggregators to process encrypted model updates while remaining unable to access plaintext values. For federated DRL, nodes encrypt local updates before transmission, and aggregators perform weighted averaging on ciphertexts, with only participating nodes possessing decryption keys. Representative works include [92, 186]. HE provides strong cryptographic privacy guarantees, protecting against malicious aggregators and eliminating trust requirements, but introduces substantial computational overhead (orders of magnitude slower than plaintext operations) that currently limits practical deployment to small models or computation-intensive phases [208].

7 Open Challenges and Future Directions

This section identifies six open challenges that emerge from the structural gaps, unresolved tensions, and unexplored intersections revealed by our two-dimensional taxonomy and four-dimensional analysis.

Challenge 1: Cross-Quadrant Migration Between SARL and MARL. Section 4.7 establishes that SARL suits moderate-scale deployments while MARL becomes necessary at larger scales, yet real-world systems do not remain at a fixed scale. No existing work addresses how to *transition* a running system between control architectures without retraining from scratch. Promising directions include progressive agent splitting that decomposes trained SARL policies into MARL agent initializations, cross-architecture knowledge distillation from global critics to per-agent critics, and hybrid architectures that dynamically adjust agent decomposition granularity. The open question is whether provable performance guarantees can be maintained during such transitions, or whether architecture switching inevitably incurs a transient penalty that must be bounded.

Challenge 2: Filling the Decentralized Federated CTDE Void. Our taxonomy identifies decentralized federated CTDE (Section 5.4.3) as a structurally empty quadrant due to an architectural incompatibility: CTDE requires centralized training components (mixing networks, centralized critics), while decentralized federation eliminates central entities. Resolving this requires fundamentally new mechanisms: gossip-based approximate mixing networks where neighbors iteratively refine local estimates of global value decomposition, hierarchical decomposition separating intra-node CTDE from inter-node federated consensus, and communication-efficient protocols that approximate centralized critics through bounded message passing. Theoretical work is needed to characterize the approximation quality and convergence properties of such distributed CTDE surrogates.

Challenge 3: The Convergence-Privacy-Overhead Triangle in Federated DRL. Table 7 repeatedly surfaces a three-way tension: stronger privacy degrades convergence and increases overhead; reducing communication exacerbates client drift; accelerating convergence increases privacy exposure. This tension is amplified in DRL by temporal correlation and policy non-stationarity, and our framework reveals that it manifests differently across quadrants: centralized federated SARL primarily faces a convergence-communication trade-off, while centralized federated MARL-CTDE must additionally protect inter-agent coordination information, expanding the attack surface. This challenge would be further compounded in decentralized federated CTDE settings (Challenge 2), where the absence of a trusted aggregator introduces additional privacy vulnerabilities through multi-hop parameter propagation. A key direction is formalizing this triangle for DRL, potentially establishing impossibility results, and developing Pareto-optimal mechanisms tailored to specific quadrant constraints.

Challenge 4: Automating Algorithm Selection Across the Taxonomy. Sections 4–5 provide extensive qualitative design guidelines, but these require expert interpretation. A natural direction is transforming them into an automated selection framework that, given observable deployment characteristics (node count, data heterogeneity, privacy requirements, resource budgets), recommends the optimal quadrant and algorithm. The search space is hierarchically structured: the first level selects the architectural configuration (SARL/MARL, standard/federated), and the second level selects the specific algorithm within that configuration. This hierarchy is compounded by prohibitive evaluation cost, as training a DRL agent to convergence requires hours to days per candidate. Promising approaches include meta-feature-based performance prediction and multi-fidelity evaluation using proxy environments. The open question is whether sufficient regularity exists across deployments to enable reliable prediction.

Challenge 5: Cross-Paradigm Benchmarking and Sim-to-Real Transfer. Our literature comparison tables 5 and 6 expose a critical gap: works across different taxonomy quadrants use incompatible evaluation settings, making cross-quadrant comparison impossible. The field requires benchmark suites that systematically vary the dimensions our taxonomy identifies, including scale gradients, heterogeneity gradients, privacy constraint levels, and inter-agent coupling strength, with standardized metrics quantifying all four analytical dimensions rather than qualitative labels. Moreover, nearly all reviewed works rely on simulation-based evaluation, leaving a significant sim-to-real gap: policies trained in simplified simulators may fail to transfer to production IoT-Edge-Cloud environments where stochastic network conditions, hardware faults, and workload burstiness deviate substantially from simulated assumptions. Developing realistic testbeds and standardized transfer protocols that validate DRL policies under real-world conditions remains an important open direction.

Challenge 6: Positioning Foundation Models Within the Taxonomy. Recent exploratory studies have applied LLMs to network optimization [83], resource orchestration [6], and scheduling [151], yet their architectural relationship to established DRL frameworks remains unclear. Three integration patterns emerge with distinct architectural implications: LLMs as *state abstractors* (operating within existing quadrants), as *meta-controllers* selecting among pre-trained DRL policies (introducing a hierarchical layer, connecting to Challenge 4), or as *direct policy generators* (potentially defining a new control paradigm). Each faces distinct challenges, namely real-time inference constraints, safety verification requirements, and lack of online adaptation, amplified in federated settings where parameter-efficient fine-tuning must be compatible with distributed aggregation [28]. Clarifying these integration patterns and their placement within the taxonomy is an important step toward principled adoption of foundation models in IoT-Edge-Cloud resource management.

8 Conclusions

In this survey, we systematically organized DRL-based resource management for the IoT-Edge-Cloud continuum through a two-dimensional taxonomy that separates control architecture from training paradigm. We comprehensively reviewed related works across single-agent and multi-agent approaches under both standard and federated training, providing a comparative analysis of DRL methods across scalability, convergence, overhead, and adaptability that reveals fundamental trade-offs and informs practical design guidelines. We also surveyed orthogonal enhancement techniques and identified open challenges with future research directions. The established frameworks and practical design guidelines advance toward intelligent, scalable, privacy-preserving resource management across the IoT-Edge-Cloud continuum.

References

[1] Sergi Abadal, Akshay Jain, Robert Guirado, Jorge López-Alonso, and Eduard Alarcón. 2021. Computing graph neural networks: A survey from algorithms to accelerators. *Comput. Surveys* 54, 9 (2021), 1–38.

[2] Siddharth Agarwal, Maria A Rodriguez, and Rajkumar Buyya. 2024. A deep recurrent-reinforcement learning method for intelligent autoscaling of serverless functions. *IEEE Transactions on Services Computing* 17, 5 (2024), 1899–1910.

[3] L. Ale, S. A. King, N. Zhang, A. R. Sattar, and J. Skandaraniyam. 2022. D3pg: Dirichlet ddpg for task partitioning and offloading with constrained hybrid action space in mobile-edge computing. *IEEE Internet of Things Journal* 9, 19 (2022), 19260–19272.

[4] Laha Ale, Ning Zhang, Xiaojie Fang, Xianfu Chen, Shaohua Wu, and Longzhuang Li. 2021. Delay-aware and energy-efficient computation offloading in mobile-edge computing using deep reinforcement learning. *IEEE Transactions on Cognitive Communications and Networking* 7, 3 (2021), 881–892.

[5] Abdulaziz Aljohani, Omer Rana, and Charith Perera. 2025. Self-adaptive federated learning in internet of things systems: A review. *Comput. Surveys* 57, 10 (2025), 1–36.

[6] Antonino Angi, Alessio Sacco, and Guido Marchetto. 2025. LLNet: An Intent-Driven Approach to Instructing Softwarized Network Devices Using a Small Language Model. *IEEE Transactions on Network and Service Management* 22, 4 (2025), 3403–3418.

[7] Ali Asghari and Mohammad Karim Sohrabi. 2024. Server placement in mobile cloud computing: A comprehensive survey for edge computing, fog computing and cloudlet. *Computer Science Review* 51 (2024), 100616.

[8] Mahmoud Assran, Joshua Romoff, Nicolas Ballas, Joelle Pineau, and Michael Rabbat. 2019. Gossip-based actor-learner architectures for deep reinforcement learning. *Advances in neural information processing systems* 32 (2019).

[9] Enrique Tomás Martínez Beltrán, Mario Quiles Pérez, Pedro Miguel Sánchez Sánchez, Sergio López Bernal, Gérôme Bovet, Manuel Gil Pérez, Gregorio Martínez Pérez, and Alberto Huertas Celdrán. 2023. Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges. *IEEE Communications Surveys & Tutorials* 25, 4 (2023), 2983–3013.

[10] Seifu Birhanu Tadele, Binayak Kar, Frezer Guteta Wakgra, and Asif Uddin Khan. 2025. Optimization of End-to-End AoI in Edge-Enabled Vehicular Fog Systems: A Dueling-DQN Approach. *IEEE Internet of Things Journal* 12, 1 (2025), 843–853.

[11] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. 2017. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security.* 1175–1191.

[12] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. 2006. Randomized gossip algorithms. *IEEE transactions on information theory* 52, 6 (2006), 2508–2530.

[13] Yue Cai, Peng Cheng, Zhuo Chen, Ming Ding, Branka Vucetic, and Yonghui Li. 2023. Deep reinforcement learning for online resource allocation in network slicing. *IEEE Transactions on Mobile Computing* 23, 6 (2023), 7099–7116.

[14] Jinwei Chai, Zhe Wang, Chuan Ma, Guanyu Gao, and Long Shi. 2025. Personalized Federated Reinforcement Learning for Multi-AAV Assisted Edge Computing. *IEEE Wireless Communications Letters* 14, 7 (2025), 2074–2078.

[15] Hengsheng Chen, Yuanguo Lin, Mingjian Fu, Lina Yao, and Michael Sheng. 2025. A Survey on Reinforcement Learning Methods for UAV Systems. *Comput. Surveys* 58, 4 (2025), 1–37.

[16] Jichao Chen, Omid Esrafilian, Harald Bayerlein, David Gesbert, and Marco Caccamo. 2023. Model-aided federated reinforcement learning for multi-UAV trajectory planning in IoT networks. In *Proceedings of the IEEE Globecom Workshops.* IEEE, 818–823.

[17] Juan Chen, Huanlai Xing, Zhiwen Xiao, Lexi Xu, and Tao Tao. 2021. A DRL agent for jointly optimizing computation offloading and resource allocation in MEC. *IEEE Internet of Things Journal* 8, 24 (2021), 17508–17524.

[18] Shijie Chen, Yu Zhang, and Qiang Yang. 2024. Multi-task learning in natural language processing: An overview. *Comput. Surveys* 56, 12 (2024), 1–32.

[19] Xianfu Chen, Celimuge Wu, Tao Chen, Zhi Liu, Honggang Zhang, Mehdi Bennis, Hang Liu, and Yusheng Ji. 2021. Information freshness-aware task offloading in air-ground integrated edge computing systems. *IEEE Journal on Selected Areas in Communications* 40, 1 (2021), 243–258.

[20] Ying Chen, Zhiyong Liu, Yongchao Zhang, Yuan Wu, Xin Chen, and Lian Zhao. 2020. Deep reinforcement learning-based dynamic resource management for mobile edge computing in industrial internet of things. *IEEE Transactions on Industrial Informatics* 17, 7 (2020), 4925–4934.

[21] Yishan Chen, Jianwei Zhang, Zhiqiang Wang, Wenshuo Dai, Honghao Gao, and Shuiguang Deng. 2025. Privacy-Preserving Knowledge Distillation in Latency-Critical Federated Task Offloading for Consumer IoT Networks. *IEEE Transactions on Consumer Electronics* 71, 1 (2025), 239–251.

[22] Zhiyong Chen, Benshun Yin, Haoyu Zhu, Yingjiao Li, Meixia Tao, and Wenjun Zhang. 2022. Mobile communications, computing, and caching resources allocation for diverse services via multi-objective proximal policy optimization. *IEEE Transactions on Communications* 70, 7 (2022), 4498–4512.

[23] Yao Chiang, Yi Zhang, Hao Luo, Tse-Yu Chen, Guan-Hao Chen, Huan-Ting Chen, Yan-Jhu Wang, Hung-Yu Wei, and Chun-Ting Chou. 2023. Management and orchestration of edge computing for IoT: A comprehensive survey. *IEEE Internet of Things Journal* 10, 16 (2023), 14307–14331.

[24] Jingjing Cui, Yuanwei Liu, and Arumugam Nallanathan. 2020. Multi-agent reinforcement learning-based resource allocation for UAV networks. *IEEE Transactions on Wireless Communications* 19, 2 (2020), 729–743.

[25] Yuya Cui, Degan Zhang, Honghu Li, Hao Qiang, and Haitao Zhao. 2026. Cooperative task offloading strategy for vehicular edge computing based on multi-agent deep reinforcement learning. *Future Generation Computer Systems* 174 (2026), 107950.

[26] Penglin Dai, Yaorong Huang, Kaiwen Hu, Xiao Wu, Huanlai Xing, and Zhaofei Yu. 2024. Meta reinforcement learning for multi-task offloading in vehicular edge computing. *IEEE Transactions on Mobile Computing* 23, 3 (2024), 2123–2138.

[27] Mohsen Darchini-Tabrizi, Amirali Pakdaman-Donyavi, Reza Entezari-Maleki, and Leonel Sousa. 2025. Performance enhancement of UAV-enabled MEC systems through intelligent task offloading and resource allocation. *Computer Networks* 264 (2025), 111280.

[28] Badhan Chandra Das, M Hadi Amini, and Yanzhao Wu. 2025. Security and privacy challenges of large language models: A survey. *Comput. Surveys* 57, 6 (2025), 1–39.

[29] Christian Schroeder De Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip HS Torr, Mingfei Sun, and Shimon Whiteson. 2020. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533* (2020).

[30] Tan Deng, Yanping Wang, Jin Li, Ronghui Cao, Yongtong Gu, Jinming Hu, Xiaoyong Tang, Mingfeng Huang, Wenzheng Liu, and Shixue Li. 2024. Entropy normalization SAC-based task offloading for UAV-assisted mobile-edge computing. *IEEE Internet of Things Journal* 11, 15 (2024), 26220–26233.

[31] Pavlos Doanis, Theodoros Giannakas, and Thrasyvoulos Spyropoulos. 2022. Scalable end-to-end slice embedding and reconfiguration based on independent DQN agents. In *Proceedings of the IEEE Global Communications Conference*. IEEE, 3429–3434.

[32] Shi Dong, Junxiao Tang, Khushnood Abbas, Ruizhe Hou, Joarder Kamruzzaman, Leszek Rutkowski, and Rajkumar Buyya. 2024. Task offloading strategies for mobile edge computing: A survey. *Computer Networks* 254 (2024), 110791.

[33] Jianbo Du, Wenjie Cheng, Guangyue Lu, Haotong Cao, Xiaoli Chu, Zhicai Zhang, and Junxuan Wang. 2021. Resource pricing and allocation in MEC enabled blockchain systems: An A3C deep reinforcement learning approach. *IEEE Transactions on Network Science and Engineering* 9, 1 (2021), 33–44.

[34] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. 2018. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *Proceedings of the International conference on machine learning*. PMLR, 1407–1416.

[35] Wentao Fan, Fan Yang, Peilong Wang, Mao Miao, Pengcheng Zhao, and Tao Huang. 2023. DRL-based service function chain edge-to-edge and edge-to-cloud joint offloading in edge-cloud network. *IEEE Transactions on Network and Service Management* 20, 4 (2023), 4478–4493.

[36] Ye Fan, Jidong Ge, Sheng Zhang, Jie Wu, and Bin Luo. 2024. Decentralized scheduling for concurrent tasks in mobile edge computing via deep reinforcement learning. *IEEE Transactions on Mobile Computing* 23, 4 (2024), 2765–2779.

[37] Mina Khoshbazm Farimani, Soroush Karimian-Aliabadi, Reza Entezari-Maleki, Bernhard Egger, and Leonel Sousa. 2024. Deadline-aware task offloading in vehicular networks using deep reinforcement learning. *Expert Systems with Applications* 249 (2024), 123622.

[38] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the International conference on machine learning*. PMLR, 1126–1135.

[39] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[40] Kaihua Fu, Wei Zhang, Quan Chen, Deze Zeng, and Minyi Guo. 2021. Adaptive resource efficient microservice deployment in cloud-edge continuum. *IEEE Transactions on Parallel and Distributed Systems* 33, 8 (2021), 1825–1840.

[41] Scott Fujimoto, Herke Hoof, and David Meger. 2018. Addressing function approximation error in actor-critic methods. In *Proceedings of the International conference on machine learning*. PMLR, 1587–1596.

[42] Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. *Advances in neural information processing systems* 29 (2016).

[43] Javier Garcıa and Fernando Fernández. 2015. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research* 16, 1 (2015), 1437–1480.

[44] Hassan Gharoun, Fereshteh Momenifar, Fang Chen, and Amir H Gandomi. 2024. Meta-learning approaches for few-shot learning: A survey of recent advances. *Comput. Surveys* 56, 12 (2024), 1–41.

[45] Yu Gong, Yifei Wei, Zhiyong Feng, F Richard Yu, and Yan Zhang. 2023. Resource allocation for integrated sensing and communication in digital twin enabled internet of vehicles. *IEEE Transactions on Vehicular Technology* 72, 4 (2023), 4510–4524.

[46] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep learning*. Vol. 1. MIT press Cambridge.

[47] Mohammad Goudarzi, Marimuthu Palaniswami, and Rajkumar Buyya. 2021. A distributed deep reinforcement learning technique for application placement in edge and fog computing environments. *IEEE Transactions on Mobile Computing* 22, 5 (2021), 2491–2505.

[48] Zhouyou Gu, Changyang She, Wibowo Hardjawana, Simon Lumb, David McKechnie, Todd Essery, and Branka Vucetic. 2021. Knowledge-assisted deep reinforcement learning in 5G scheduler design: From theoretical framework to implementation. *IEEE Journal on Selected Areas in Communications* 39, 7 (2021), 2014–2028.

[49] Jinsong Gui and Hanjian Liu. 2025. Data-Driven Resource Allocation for Ensuring Remote Data Collection Timeliness in Integrated Ground-Air-Space Networks. *Computer Networks* (2025), 111715.

[50] Liang Guo, Chen-Khong Tham, Jie Jia, Jian Chen, and Xingwei Wang. 2025. Generative diffusion model-based QMIX for joint task offloading and resource allocation in VEC systems. *Computer Networks* (2025), 111516.

[51] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the International conference on machine learning*. PMLR, 1861–1870.

[52] Rui Han, Shilin Wen, Chi Harold Liu, Ye Yuan, Guoren Wang, and Lydia Y Chen. 2022. EdgeTuner: Fast scheduling algorithm tuning for dynamic edge-cloud workloads and resources. In *Proceedings of the IEEE Conference on Computer Communications*. IEEE, 880–889.

[53] Hao Hao, Changqiao Xu, Wei Zhang, Xingyan Chen, Shujie Yang, and Gabriel-Miro Muntean. 2025. Reliability-Aware Optimization of Task Offloading for UAV-Assisted Edge Computing. *IEEE Trans. Comput.* 74, 11 (2025), 3832–3844.

[54] Xin Hao, Phee Lep Yeoh, Changyang She, Branka Vucetic, and Yonghui Li. 2024. Secure deep reinforcement learning for dynamic resource allocation in wireless MEC networks. *IEEE Transactions on Communications* 72, 3 (2024), 1414–1427.

[55] Yixue Hao, Min Chen, Hamid Gharavi, Yin Zhang, and Kai Hwang. 2020. Deep reinforcement learning for edge service placement in softwarized industrial cyber-physical system. *IEEE Transactions on Industrial Informatics* 17, 8 (2020), 5552–5561.

[56] Abolfazl Hashemi, Anish Acharya, Rudrajit Das, Haris Vikalo, Sujay Sanghavi, and Inderjit Dhillon. 2021. On the benefits of multiple gossip steps in communication-constrained decentralized federated learning. *IEEE Transactions on Parallel and Distributed Systems* 33, 11 (2021), 2727–2739.

[57] Haoran He, Fanqin Zhou, Yikun Zhao, Wenjing Li, and Lei Feng. 2023. Hypergraph convolution mix DDPG for multi-aerial base station deployment. *Journal of Cloud Computing* 12, 1 (2023), 172.

[58] Qiang He, Yang Xia, Zheng Feng, Lianbo Ma, Yingjie Lv, Keping Yu, Ammar Hawbani, Kaifa Zheng, and Li Xu. 2026. Computation Resource Management in Mobile Edge Computing for Healthcare Using Lyapunov-Deep Deterministic Policy Gradient. *IEEE Transactions on Mobile Computing* 25, 2 (2026), 2159–2171.

[59] Xiaoming He, Yunzhe Jiang, Huajun Cui, Yinqiu Liu, Mingkai Chen, Maher Guizani, and Shahid Mumtaz. 2025. QoE-Driven Proactive Caching With DRL in Sustainable Cloud-to-Edge Continuum. *IEEE Transactions on Mobile Computing* 24, 10 (2025), 10992–11004.

[60] Ali Reza Heidarpour, Mohammad Reza Heidarpour, Masoud Ardakani, Chintha Tellambura, and Murat Uysal. 2023. Soft actor–critic-based computation offloading in multiuser MEC-enabled IoT—A lifetime maximization perspective. *IEEE Internet of Things Journal* 10, 20 (2023), 17571–17584.

[61] Pablo Hernandez-Leal, Bilal Kartal, and Matthew E Taylor. 2019. A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems* 33, 6 (2019), 750–797.

[62] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. 2018. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[63] Dan Horgan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado van Hasselt, and David Silver. 2018. Distributed Prioritized Experience Replay. In *Proceedings of the International Conference on Learning Representations*.

[64] Diego Hortelano, Ignacio de Miguel, Ramón J Durán Barroso, Juan Carlos Aguado, Noemí Merayo, Lidia Ruiz, Adrian Asensio, Xavi Masip-Bruin, Patricia Fernández, Rubén M Lorenzo, et al. 2023. A comprehensive survey on reinforcement-learning-based computation offloading techniques in edge computing systems. *Journal of Network and Computer Applications* 216 (2023), 103669.

[65] Peng Hou, Yi Huang, Hongbin Zhu, Zhihui Lu, Shin-Chia Huang, and Hongfeng Chai. 2024. Intelligent decision-based edge server sleep for green computing in MEC-enabled IoV networks. *IEEE Transactions on Intelligent Vehicles* 9, 2 (2024), 3687–3703.

[66] Bintao Hu, Wenzhang Zhang, Yuan Gao, Jianbo Du, and Xiaoli Chu. 2024. Multiagent deep deterministic policy gradient-based computation offloading and resource allocation for ISAC-aided 6G V2X networks. *IEEE Internet of Things Journal* 11, 20 (2024), 33890–33902.

[67] Shi-Hong Hu, Qu-Yuan Luo, Guang-Hui Li, Weisong Shi, and Bao-Liu Ye. 2023. Ca-dts: A distributed and collaborative task scheduling algorithm for edge computing enabled intelligent road network. *Journal of Computer Science and Technology* 38, 5 (2023), 1113–1131.

[68] Congyue Huang, Wei Tan, Miaohua Ou, Erfu Yang, and Yun Li. 2026. KMPS: A Reinforcement Learning Scheduler for Kubernetes Edge-Cloud Systems. *IEEE Internet of Things Journal* (2026), 1–1.

[69] Jaewon Jeong and Joohyung Lee. 2024. A Federated Reinforcement Learning Framework via a Committee Mechanism for Resource Management in 5G Networks. *Sensors* 24, 21 (2024), 7031.

[70] Zelin Ji, Zhijin Qin, Xiaoming Tao, and Zhu Han. 2024. Resource optimization for semantic-aware networks with task offloading. *IEEE Transactions on Wireless Communications* 23, 9 (2024), 12284–12296.

[71] Feibo Jiang, Yubo Peng, Kezhi Wang, Li Dong, and Kun Yang. 2023. MARS: A DRL-based multi-task resource scheduling framework for UAV with IRS-assisted mobile edge computing system. *IEEE Transactions on Cloud Computing* 11, 4 (2023), 3700–3712.

[72] Shurui Jiang, Jun Zheng, Feng Yan, and Shuyuan Zhao. 2023. Reinforcement-learning-based network slicing and resource allocation for multi-access edge computing networks. *IEEE Transactions on Cognitive Communications and Networking* 10, 3 (2023), 1132–1145.

[73] Julian Jimenez, Paola Soto, Danny De Vleeschauwer, Chia-Yu Chang, Yorick De Bock, Steven Latre, and Miguel Camelo. 2023. Resource allocation of multi-user workloads in cloud and edge data-centers using reinforcement learning. In *Proceedings of the International Conference on Network and Service Management*. IEEE, 1–5.

[74] Jiangliang Jin and Yunjian Xu. 2021. Optimal policy characterization enhanced proximal policy optimization for multitask scheduling in cloud computing. *IEEE Internet of Things Journal* 9, 9 (2021), 6418–6433.

[75] Lei Jin, Junyan Chen, Rui Yao, Jiahao Chen, and Xinmei Li. 2025. SEROS: Shared exploration and reward optimization task scheduling strategy for multi-agent collaboration in edge computing networks. *Ad Hoc Networks* (2025), 103992.

[76] Ying Ju, Zhiwei Cao, Yuchao Chen, Lei Liu, Qingqi Pei, Shahid Mumtaz, Mianxiong Dong, and Mohsen Guizani. 2023. NOMA-assisted secure offloading for vehicular edge computing networks with asynchronous deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems* 25, 3 (2023), 2627–2640.

[77] Hongyue Kang, Xiaolin Chang, Jelena Mišić, Vojislav B Mišić, Junchao Fan, and Yating Liu. 2023. Cooperative UAV resource allocation and task offloading in hierarchical aerial computing systems: A MAPPO-based approach. *IEEE Internet of Things Journal* 10, 12 (2023), 10497–10509.

[78] Steven Kapturowski, Georg Ostrovski, John Quan, Remi Munos, and Will Dabney. 2018. Recurrent experience replay in distributed reinforcement learning. In *Proceedings of the International conference on learning representations*.

[79] Anastasia Koloskova, Nicolas Loizou, Sadra Boreiri, Martin Jaggi, and Sebastian Stich. 2020. A unified theory of decentralized SGD with changing topology and local updates. In *Proceedings of the International conference on machine learning*. PMLR, 5381–5393.

[80] Lingjing Kong, Tao Lin, Anastasia Koloskova, Martin Jaggi, and Sebastian Stich. 2021. Consensus control for decentralized deep learning. In *Proceedings of the International Conference on Machine Learning*. PMLR, 5686–5696.

[81] Linghe Kong, Jinlin Tan, Junqin Huang, Guihai Chen, Shuaitian Wang, Xi Jin, Peng Zeng, Muhammad Khan, and Sajal K Das. 2022. Edge-computing-driven internet of things: A survey. *Comput. Surveys* 55, 8 (2022), 1–41.

[82] Matthew Landers and Afsaneh Doryab. 2023. Deep reinforcement learning verification: A survey. *Comput. Surveys* 55, 14s (2023), 1–31.

[83] Hoon Lee, Wentao Zhou, Mérouane Debbah, and Inkyu Lee. 2025. On the Convergence of Large Language Model Optimizer for Black-Box Network Management. *IEEE Transactions on Communications* 73, 11 (2025), 11385–11402.

[84] Songxin Lei, Huijun Tang, Chuangyi Li, Xueying Zhang, Chenli Xu, and Huaming Wu. 2026. Federated MADDPG-Based Collaborative Scheduling Strategy in Vehicular Edge Computing. *IEEE Transactions on Mobile Computing* 25, 1 (2026), 54–66.

[85] Chunlin Li, Kun Jiang, Guangxuan He, Fan Bing, and Youlong Luo. 2024. A computation offloading method for multi-UAVs assisted MEC based on improved federated DDPG algorithm. *IEEE Transactions on Industrial Informatics* 20, 12 (2024), 14062–14071.

[86] Chunlin Li, Shuai Zhang, Yaojuan Wu, Kun Jiang, Wenhao Wu, Shaochong Yuan, and Shaohua Wan. 2025. Improved TD3 Based Resource Allocation Optimization for Latency-sensitive Tasks in ISAC-aided VEC. *ACM Transactions on Sensor Networks* 21, 6 (2025), 1–27.

[87] Huan Li and Zhouchen Lin. 2024. Accelerated gradient tracking over time-varying graphs for decentralized optimization. *Journal of Machine Learning Research* 25, 274 (2024), 1–52.

[88] Han Li, Ke Xiong, Yuping Lu, Wei Chen, Pingyi Fan, and Khaled Ben Letaief. 2024. Collaborative task offloading and resource allocation in small-cell MEC: A multi-agent PPO-based scheme. *IEEE Transactions on Mobile Computing* 24, 3 (2024), 2346–2359.

[89] Meng Li, Sixing Ma, Pengbo Si, and Haijun Zhang. 2024. Relay selection and resource allocation for ad hoc networks-assisted train-to-train communications: A federated soft actor-critic approach. *IEEE Transactions on Vehicular Technology* 73, 10 (2024), 15359–15371.

[90] Suyi Li, Luping Wang, Wei Wang, Yinghao Yu, and Bo Li. 2021. George: Learning to place long-lived containers in large clusters with operation constraints. In *Proceedings of the ACM Symposium on Cloud Computing*. 258–272.

[91] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. *Proceedings of the Machine learning and systems* 2 (2020), 429–450.

[92] Xuehan Li, Tao Jing, Ruinian Li, Xiaoxuan Wang, Yu Yan, Xin Fan, Yan Huo, and Fei Richard Yu. 2025. Self-Learning Based Dependable Offloading Optimization in Semi-Trusted Vehicular Edge Computing and Networks. *IEEE Transactions on Vehicular Technology* 74, 5 (2025), 8141–8157.

[93] Yuchen Li, Weifa Liang, and Jing Li. 2022. Profit driven service provisioning in edge computing via deep reinforcement learning. *IEEE Transactions on Network and Service Management* 19, 3 (2022), 3006–3019.

[94] Yihong Li, Xiaoxi Zhang, Tianyu Zeng, Jingpu Duan, Chuan Wu, Di Wu, and Xu Chen. 2023. Task placement and resource allocation for edge machine learning: A gnn-based multi-agent reinforcement learning paradigm. *IEEE Transactions on Parallel and Distributed Systems* 34, 12 (2023), 3073–3089.

[95] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. 2017. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *Advances in neural information processing systems* 30 (2017).

[96] Yonghui Liang, Huijun Tang, Huaming Wu, Yixiao Wang, and Pengfei Jiao. 2024. Lyapunov-guided offloading optimization based on soft actor-critic for isac-aided internet of vehicles. *IEEE Transactions on Mobile Computing* 23, 12 (2024), 14708–14721.

[97] Linbo Liao, Yongxuan Lai, Fan Yang, and Wenhua Zeng. 2023. Online computation offloading with double reinforcement learning algorithm in mobile edge computing. *J. Parallel and Distrib. Comput.* 171 (2023), 28–39.

[98] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2016. Continuous control with deep reinforcement learning. In *Proceedings of the International Conference on Learning Representations*.

[99] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and Bill Dally. 2018. Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training. In *Proceedings of the International Conference on Learning Representations*.

[100] Yan Lin, Liqin Xiao, Yiyu Tao, Yijin Zhang, Feng Shu, and Jun Li. 2025. Multi-Agent Computing-Energy-Efficiency Optimization in Vehicular Edge Computing: Non-Cooperative Versus Cooperative Solutions. *IEEE Transactions on Wireless Communications* 24, 7 (2025), 5461–5476.

[101] Chubo Liu, Fan Tang, Yikun Hu, Kenli Li, Zhuo Tang, and Keqin Li. 2021. Distributed task migration optimization in MEC by extending multi-agent deep reinforcement learning approach. *IEEE Transactions on Parallel and Distributed Systems* 32, 7 (2021), 1603–1614.

[102] Chi Harold Liu, Zipeng Dai, Yinuo Zhao, Jon Crowcroft, Dapeng Wu, and Kin K Leung. 2021. Distributed and energy-efficient mobile crowdsensing with charging stations by deep reinforcement learning. *IEEE Transactions on Mobile Computing* 20, 1 (2021), 130–146.

[103] Pengxue Liu, Dalong Zhang, Fasong Wang, Shijie Shi, Yanbin Zhang, and Yitong Li. 2025. Priority-Aware Resource Allocation in AoI-Oriented UL-OFDMA Wi-Fi Networks Based on Multiagent Reinforcement Learning. *IEEE Internet of Things Journal* 12, 20 (2025), 42045–42058.

[104] Qingli Liu and Yongjie Ma. 2025. Communication resource allocation method in vehicular networks based on federated multi-agent deep reinforcement learning. *Scientific Reports* 15, 1 (2025), 30866.

[105] Xiaoying Liu, Anping Chen, Kechen Zheng, Kaikai Chi, Bin Yang, and Tarik Taleb. 2024. Distributed computation offloading for energy provision minimization in WP-MEC networks with multiple HAPs. *IEEE Transactions on Mobile Computing* 24, 4 (2024), 2673–2689.

[106] Zhang Liu, Lianfen Huang, Zhibin Gao, Manman Luo, Seyyedali Hosseinalipour, and Huaiyu Dai. 2024. GA-DRL: Graph neural network-augmented deep reinforcement learning for DAG task scheduling over dynamic vehicular clouds. *IEEE Transactions on Network and Service Management* 21, 4 (2024), 4226–4242.

[107] Zijian Long, Haopeng Wang, Haiwei Dong, and Abdulmotaleb El Saddik. 2025. Adaptive Social Metaverse Streaming Based on Federated Multiagent Deep Reinforcement Learning. *IEEE Transactions on Computational Social Systems* 12, 5 (2025), 3804–3815.

[108] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems* 30 (2017).

[109] Jialin Lu, Jing Yang, Shaobo Li, Yijun Li, Wu Jiang, Jiangtian Dai, and Jianjun Hu. 2024. A2C-DRL: Dynamic Scheduling for Stochastic Edge–Cloud Environments Using A2C and Deep Reinforcement Learning. *IEEE Internet of Things Journal* 11, 9 (2024), 16915–16927.

[110] Xiaozhen Lu, Liang Xiao, Yilin Xiao, Zehui Xiong, Zhe Liu, Yanyong Zhang, and Weihua Zhuang. 2025. Blockchain-Enabled Secure Offloading for VEC: A Multi-Agent Reinforcement Learning Approach. *IEEE Transactions on Dependable and Secure Computing* 22, 3 (2025), 2978–2995.

[111] Xueguang Lyu, Andrea Baisero, Yuchen Xiao, Brett Daley, and Christopher Amato. 2023. On centralized critics in multi-agent reinforcement learning. *Journal of Artificial Intelligence Research* 77 (2023), 295–354.

[112] Bjarke Madsen and Ramoni Adeogun. 2024. Federated multi-agent drl for radio resource management in industrial 6g in-x subnetworks. In *Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE, 1–7.

[113] Adyson Maia, Akram Boutouchent, Youcef Kardjadja, Manel Gherari, Ece Gelal Soyak, Muhammad Saqib, Kacem Boussekar, Idil Cilbir, Sama Habibi, Soukaina Ouledsidi Ali, et al. 2024. A survey on integrated computing, caching, and communication in the cloud-to-edge continuum. *Computer Communications* 219 (2024), 128–152.

[114] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.

[115] Gaurav Menghani. 2023. Efficient deep learning: A survey on making deep learning models smaller, faster, and better. *Comput. Surveys* 55, 12 (2023), 1–37.

[116] Stephanie Milani, Nicholay Topin, Manuela Veloso, and Fei Fang. 2024. Explainable reinforcement learning: A survey and comparative review. *Comput. Surveys* 56, 7 (2024), 1–36.

[117] Kaushik Mishra, Goluguri NV Rajareddy, Umashankar Ghugar, Gurpreet Singh Chhabra, and Amir H Gandomi. 2023. A collaborative computation and offloading for compute-intensive and latency-sensitive dependency-aware tasks in dew-enabled vehicular fog computing: A federated deep Q-learning approach. *IEEE Transactions on Network and Service Management* 20, 4 (2023), 4600–4614.

[118] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *Proceedings of the International conference on machine learning*. PmLR, 1928–1937.

[119] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature* 518, 7540 (2015), 529–533.

[120] Komeil Moghaddasi and Raja Jurdak. 2026. An energy-aware distributed federated soft actor-critic framework for intelligent task offloading in vehicular mobile edge computing networks. *Ad Hoc Networks* (2026), 104043.

[121] Hoa T Nguyen, Muhammad Usman, and Rajkumar Buyya. 2024. Drlq: A deep reinforcement learning-based task placement for quantum cloud computing. In *Proceedings of the IEEE International Conference on Cloud Computing*. IEEE, 475–481.

[122] Vahidreza Niazmand and Qiang Ye. 2025. Joint Task Offloading, DNN Pruning, and Computing Resource Allocation for Fault Detection With Dynamic Constraints in Industrial IoT. *IEEE Transactions on Cognitive Communications and Networking* 11, 5 (2025), 3486–3501.

[123] Farnaz Niknia and Ping Wang. 2025. Edge Caching Optimization With PPO and Transfer Learning for Dynamic Environments. *IEEE Internet of Things Journal* 12, 16 (2025), 33605–33620.

[124] Pengfei Ning, Hongwei Wang, Tao Tang, Jie Zhang, Hongyang Du, Dusit Niyato, and F Richard Yu. 2025. Diffusion-based deep reinforcement learning for resource management in connected construction equipment networks: A hierarchical framework. *IEEE Transactions on Wireless Communications* 24, 4 (2025), 2847–2861.

[125] Chao Pan, Zhao Wang, Haijun Liao, Zhenyu Zhou, Xiaoyan Wang, Muhammad Tariq, and Sattam Al-Otaibi. 2023. Asynchronous federated deep reinforcement learning-based URLLC-aware computation offloading in space-assisted vehicular networks. *IEEE Transactions on Intelligent Transportation Systems* 24, 7 (2023), 7377–7389.

[126] Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V Albrecht. 2021. Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks. In *Proceedings of the Conference on Neural Information Processing Systems*.

[127] Haixia Peng and Xuemin Shen. 2021. Multi-agent reinforcement learning based resource management in MEC-and UAV-assisted vehicular networks. *IEEE Journal on Selected Areas in Communications* 39, 1 (2021), 131–141.

[128] Nuoheng Peng, Yan Lin, Yijin Zhang, and Jun Li. 2022. Aoi-aware joint spectrum and power allocation for internet of vehicles: A trust region policy optimization-based approach. *IEEE Internet of Things Journal* 9, 20 (2022), 19916–19927.

[129] Yan Peng, Xiaogang Tang, Yiqing Zhou, Jintao Li, Yanli Qi, Ling Liu, and Hai Lin. 2024. Computing and communication cost-aware service migration enabled by transfer reinforcement learning for dynamic vehicular edge computing networks. *IEEE Transactions on Mobile Computing* 23, 1 (2024), 257–269.

[130] Kilian Pfeiffer, Martin Rapp, Ramin Khalili, and Jörg Henkel. 2023. Federated learning for computationally constrained heterogeneous devices: A survey. *Comput. Surveys* 55, 14s (2023), 1–27.

[131] Priyadarshni, Dhruvan Kadavala, Shivani Tripathi, Praveen Kumar, and Rajiv Misra. 2025. Deep meta reinforcement learning for efficient task offloading in edge computing environments. *Cluster Computing* 28, 11 (2025), 712.

[132] Langtian Qin, Hancheng Lu, Yuang Chen, Baolin Chong, and Feng Wu. 2024. Toward Decentralized Task Offloading and Resource Allocation in User-Centric MEC. *IEEE Transactions on Mobile Computing* 23, 12 (2024), 11807–11823.

[133] Yeguang Qin, Yilin Yang, Fengxiao Tang, Xin Yao, Ming Zhao, and Nei Kato. 2024. Differentiated federated reinforcement learning based traffic offloading on space-air-ground integrated networks. *IEEE Transactions on Mobile Computing* 23, 12 (2024), 11000–11013.

[134] Guannan Qu and Na Li. 2017. Harnessing smoothness to accelerate distributed optimization. *IEEE Transactions on Control of Network Systems* 5, 3 (2017), 1245–1260.

[135] Youyang Qu, Md Palash Uddin, Chenquan Gan, Yong Xiang, Longxiang Gao, and John Yearwood. 2022. Blockchain-enabled federated learning: A survey. *Comput. Surveys* 55, 4 (2022), 1–35.

[136] Asif Mahmud Raivi and Sangman Moh. 2024. JDACO: Joint Data Aggregation and Computation Offloading in UAV-Enabled Internet of Things for Post-Disaster Scenarios. *IEEE Internet of Things Journal* 11, 9 (2024), 16529–16544.

[137] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2018. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning*. PMLR, 4295–4304.

[138] Nayereh Rasouli, Cristian Klein, and Erik Elmroth. 2025. Resource management for mission-critical applications in edge computing: systematic review on recent research and open issues. *Comput. Surveys* 58, 3 (2025), 1–37.

[139] Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. 2021. Adaptive Federated Optimization. In *Proceedings of the International Conference on Learning Representations*.

[140] Yin Ren, Suhao Yu, and Aihuang Guo. 2026. Joint Scheduling Mechanism for Dynamic Slice Resource Allocation and Task Offloading in User-Edge-Cloud Systems. *IEEE Transactions on Cloud Computing* (2026), 1–17.

[141] Desik Rengarajan, Nitin Ragothaman, Dileep Kalathil, and Srinivas Shakkottai. 2024. Federated ensemble-directed offline reinforcement learning. *Advances in Neural Information Processing Systems* 37 (2024), 6154–6179.

[142] Shavbo Salehi, Pedro Enrique Iturria-Rivera, Medhat Elsayed, Majid Bavand, Raimundas Gaigalas, Yigit Ozcan, and Melike Erol-Kantarci. 2025. Prioritized Value-Decomposition Network for Explainable AI-Enabled Network Slicing. In *Proceedings of the IEEE International Conference on Communications*. IEEE, 572–577.

[143] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *Proceedings of the International conference on machine learning*. PMLR, 1889–1897.

[144] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[145] Nelson Sharma, Aswini Ghosh, Rajiv Misra, and Sajal K Das. 2024. Deep meta q-learning based multi-task offloading in edge-cloud systems. *IEEE Transactions on Mobile Computing* 23, 4 (2024), 2583–2598.

[146] Si Shen, Guojiang Shen, Zhehao Dai, Kaiyu Zhang, Xiangjie Kong, and Jianxin Li. 2024. Asynchronous Federated Deep-Reinforcement-Learning-Based Dependency Task Offloading for UAV-Assisted Vehicular Networks. *IEEE Internet of Things Journal* 11, 19 (2024), 31561–31574.

[147] Rana Muhammad Sohaib, Oluwakayode Onireti, Yusuf Sambo, Rafiq Swash, and Muhammad Imran. 2024. Energy efficient resource allocation framework based on dynamic meta-transfer learning for V2X communications. *IEEE Transactions on Network and Service Management* 21, 4 (2024), 4343–4356.

[148] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. 2019. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *Proceedings of the International conference on machine learning*. PMLR, 5887–5896.

[149] Fazal E Subhan, Abid Yaqoob, Cristina Hava Muntean, and Gabriel-Miro Muntean. 2025. EDGE360: Edge-Enabled Multi-Agent DRL for Region-Aware Rate Adaptation Solution to Enhance Quality of 360° Video Streaming. *IEEE Transactions on Mobile Computing* (2025), 1–18.

[150] Chuan Sun, Xiongwei Wu, Xiuhua Li, Qilin Fan, Junhao Wen, and Victor CM Leung. 2021. Cooperative computation offloading for multi-access edge computing in 6G mobile networks via soft actor critic. *IEEE Transactions on Network Science and Engineering* 11, 6 (2021), 5601–5614.

[151] Minghao Sun, Jinbo Hou, Kehai Qiu, Kezhi Wang, Xiaoli Chu, and Zitian Zhang. 2025. LLM-based Task Offloading and Resource Allocation in Satellite Edge Computing Networks. *IEEE Transactions on Vehicular Technology* (2025), 1–6.

[152] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. 2018. Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward. In *Proceedings of the International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2085–2087.

[153] Richard S Sutton, Andrew G Barto, et al. 1998. *Reinforcement learning: An introduction*. Vol. 1. MIT press Cambridge.

[154] Imane Taleb, Jean-Loup Guillaume, and Benjamin Duthil. 2025. A Survey on Services Placement Algorithms in Integrated Cloud-Fog/Edge Computing. *Comput. Surveys* 57, 11 (2025), 1–36.

[155] Ardi Tampuu, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente. 2017. Multiagent cooperation and competition with deep reinforcement learning. *PloS one* 12, 4 (2017), e0172395.

[156] Chaogang Tang, Yao Ding, Shuo Xiao, Zhenzhen Huang, and Huaming Wu. 2025. Collaborative Service Caching, Task Offloading, and Resource Allocation in Caching-Assisted Mobile Edge Computing. *IEEE Transactions on Services Computing* 18, 4 (2025), 1966–1981.

[157] Qinqin Tang, Renchao Xie, Fei Richard Yu, Tianjiao Chen, Ran Zhang, Tao Huang, and Yunjie Liu. 2022. Collective deep reinforcement learning for intelligence sharing in the internet of intelligence-empowered edge computing. *IEEE Transactions on Mobile Computing* 22, 11 (2022), 6327–6342.

[158] Shujiong Tang, Yue Yu, Hui Wang, Guiliang Wang, Wuhui Chen, Zenglin Xu, Song Guo, and Wen Gao. 2024. A survey on scheduling techniques in computing and network convergence. *IEEE Communications Surveys & Tutorials* 26, 1 (2024), 160–195.

[159] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2022. Efficient Transformers: A Survey. *Comput. Surveys* 55, 6 (2022).

[160] Belayneh Abebe Tesfaw, Rong-Terng Juang, Getaneh Berie Tarekegn, Wendenda Nathanael Kabore, and Ming-Cheng Tsai. 2025. Joint UAV 3-D Trajectory and Resource Allocation for Integrated LEO Satellite and Multi-UAV-Enabled Marine IoT Networks: A Federated Multiagent Deep Reinforcement Learning Approach. *IEEE Internet of Things Journal* 12, 21 (2025), 45076–45093.

[161] Bingxin Tian, Li Wang, Lianming Xu, Wen Pan, Huaqing Wu, Liang Li, and Zhu Han. 2023. UAV-assisted wireless cooperative communication and coded caching: A multiagent two-timescale DRL approach. *IEEE Transactions on Mobile Computing* 23, 5 (2023), 4389–4404.

[162] Zhao Tong, Jiaxin Deng, Jing Mei, Yuanyang Zhang, and Keqin Li. 2024. Multi-Objective DAG Task Offloading in MEC Environment Based on Federated DQN With Automated Hyperparameter Optimization. *IEEE Transactions on Services Computing* 17, 6 (2024), 3999–4012.

[163] Claudia Torres-Pérez, Estefanía Coronado, Cristina Cervelló-Pastor, Javier Palomares, Estela Carmona-Cejudo, and Muhammad Shuaib Siddiqui. 2025. Minimizing active nodes in MEC environments: A distributed learning-driven framework for application placement. *Computer Networks* 257 (2025), 111008.

[164] Shreshth Tuli, Shashikant Ilager, Kotagiri Ramamohanarao, and Rajkumar Buyya. 2020. Dynamic scheduling for stochastic edge-cloud computing environments using a3c learning and residual recurrent neural networks. *IEEE transactions on mobile computing* 21, 3 (2020), 940–954.

[165] Hado Van Hasselt, Arthur Guez, and David Silver. 2016. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 30.

[166] Divya Vetriveeran and Leena Sri R. 2025. Resource Provisioning in Fog Computing - A Survey. *Comput. Surveys* 57, 12 (2025).

[167] Guneet Kaur Walia, Mohit Kumar, and Sukhpal Singh Gill. 2024. AI-empowered fog/edge resource management for IoT applications: A comprehensive review, research challenges, and future perspectives. *IEEE Communications Surveys & Tutorials* 26, 1 (2024), 619–669.

[168] Guangchen Wang, Peng Cheng, Zhuo Chen, Branka Vucetic, and Yonghui Li. 2024. Inverse reinforcement learning with graph neural networks for full-dimensional task offloading in edge computing. *IEEE Transactions on Mobile Computing* 23, 6 (2024), 6490–6507.

[169] Jin Wang, Jia Hu, Geyong Min, Albert Y. Zomaya, and Nektarios Georgalas. 2021. Fast Adaptive Task Offloading in Edge Computing Based on Meta Reinforcement Learning. *IEEE Transactions on Parallel and Distributed Systems* 32, 1 (2021), 242–253.

[170] Lei Wang, Weixi Mao, Jin Zhao, and Yuedong Xu. 2021. DDQP: A double deep Q-learning approach to online fault-tolerant SFC placement. *IEEE Transactions on Network and Service Management* 18, 1 (2021), 118–132.

[171] Lun Wang, Yang Xu, Hongli Xu, Min Chen, and Liusheng Huang. 2022. Accelerating decentralized federated learning in heterogeneous edge computing. *IEEE Transactions on Mobile Computing* 22, 9 (2022), 5001–5016.

[172] Xiaojie Wang, Jiameng Li, Zhaolong Ning, Qingyang Song, Lei Guo, Song Guo, and Mohammad S Obaidat. 2023. Wireless powered mobile edge computing networks: A survey. *Comput. Surveys* 55, 13s (2023), 1–37.

[173] Xubin Wang, Zhiqing Tang, Jianxiong Guo, Tianhui Meng, Chenhao Wang, Tian Wang, and Weijia Jia. 2025. Empowering edge intelligence: A comprehensive survey on on-device ai models. *Comput. Surveys* 57, 9 (2025), 1–39.

[174] Zhiyu Wang, Mohammad Goudarzi, and Rajkumar Buyya. 2025. ReinFog: A Deep Reinforcement Learning empowered framework for resource management in edge and cloud computing environments. *Journal of Network and Computer Applications* 242 (2025), 104250.

[175] Zhiyu Wang, Mohammad Goudarzi, and Rajkumar Buyya. 2025. TF-DDRL: A Transformer-Enhanced Distributed DRL Technique for Scheduling IoT Applications in Edge and Cloud Computing Environments. *IEEE Transactions on Services Computing* 18, 2 (2025), 1039–1053.

[176] Zhiyu Wang, Mohammad Goudarzi, Mingming Gong, and Rajkumar Buyya. 2024. Deep reinforcement learning-based scheduling for optimizing system load and response time in edge and fog computing environments. *Future Generation Computer Systems* 152 (2024), 55–69.

[177] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. 2016. Dueling network architectures for deep reinforcement learning. In *Proceedings of the International conference on machine learning*. PMLR, 1995–2003.

[178] Noor Waqar, Syed Ali Hassan, Haris Pervaiz, Haejoon Jung, and Kapal Dev. 2022. Deep multi-agent reinforcement learning for resource allocation in NOMA-enabled MEC. *Computer Communications* 196 (2022), 1–8.

[179] Zhiwei Wei, Jingxin Mao, Bing Li, and Rongqing Zhang. 2025. Privacy-Preserving Hierarchical Reinforcement Learning Framework for Task Offloading in Low-Altitude Vehicular Fog Computing. *IEEE Open Journal of the Communications Society* 6 (2025), 3389–3403.

[180] Jiin Woo, Gauri Joshi, and Yuejie Chi. 2025. The blessing of heterogeneity in federated q-learning: Linear speedup and beyond. *Journal of Machine Learning Research* 26, 26 (2025), 1–85.

[181] Huaming Wu, Lei Tian, Huijun Tang, Ruidong Li, and Pengfei Jiao. 2025. Graph Convolutional Reinforcement Learning-Guided Joint Trajectory Optimization and Task Offloading for Aerial Edge Computing. *IEEE Transactions on Intelligent Transportation Systems* 26, 10 (2025), 17487–17498.

[182] Jiajun Wu, Fan Dong, Henry Leung, Zhuangdi Zhu, Jiayu Zhou, and Steve Drew. 2024. Topology-aware federated learning in edge computing: A comprehensive survey. *Comput. Surveys* 56, 10 (2024), 1–41.

[183] Yujian Wu, Shanjiang Tang, Ce Yu, Bin Yang, Chao Sun, Jian Xiao, Hutong Wu, and Jinghua Feng. 2025. Task Scheduling in Geo-Distributed Computing: A Survey. *IEEE Transactions on Parallel and Distributed Systems* 36, 10 (2025), 2073–2088.

[184] Renchao Xie, Li Feng, Qinqin Tang, Tao Huang, Zehui Xiong, Tianjiao Chen, and Ran Zhang. 2024. Delay-Prioritized and Reliable Task Scheduling With Long-Term Load Balancing in Computing Power Networks. *IEEE Transactions on Services Computing* 17, 6 (2024), 3359–3372.

[185] Xiong Xiong, Kan Zheng, Lei Lei, and Lu Hou. 2020. Resource allocation based on deep reinforcement learning in IoT edge computing. *IEEE Journal on Selected Areas in Communications* 38, 6 (2020), 1133–1146.

[186] Jianbin Xue, Jia Yao, and Jiahao Wang. 2024. A dynamic pricing scheme for secure offloading and resource allocation based on the internet of vehicles. *Ad Hoc Networks* 161 (2024), 103545.

[187] Zeyue Xue, Pan Zhou, Zichuan Xu, Xiumin Wang, Yulai Xie, Xiaofeng Ding, and Shiping Wen. 2021. A resource-constrained and privacy-preserving edge-computing-enabled clinical decision system: A federated reinforcement learning approach. *IEEE Internet of Things Journal* 8, 11 (2021), 9122–9138.

[188] Jing Yang, Jialin Lu, Xu Zhou, Shaobo Li, Chuanyue Xiong, and Jianjun Hu. 2024. HA-A2C: Hard attention and advantage actor-critic for addressing latency optimization in edge computing. *IEEE Transactions on Green Communications and Networking* 9, 1 (2024), 207–217.

[189] Ning Yang, Shuo Chen, Haijun Zhang, and Randall Berry. 2025. Beyond the edge: An advanced exploration of reinforcement learning for mobile edge computing, its applications, and future research trajectories. *IEEE Communications Surveys & Tutorials* 27, 1 (2025), 546–594.

[190] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology* 10, 2 (2019), 1–19.

[191] Tong Yang, Shicong Cen, Yuting Wei, Yuxin Chen, and Yuejie Chi. 2024. Federated natural policy gradient and actor critic methods for multi-task reinforcement learning. *Advances in Neural Information Processing Systems* 37 (2024), 121304–121375.

[192] Liang Yao, Xiaolong Xu, Muhammad Bilal, and Huihui Wang. 2022. Dynamic edge computation offloading for internet of vehicles with deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems* 24, 11 (2022), 12991–12999.

[193] Su Yao, Mu Wang, Ju Ren, Tianyu Xia, Weiqiang Wang, Ke Xu, Mingwei Xu, and Hongke Zhang. 2025. Multi-Agent Reinforcement Learning for Task Offloading in Crowd-Edge Computing. *IEEE Transactions on Mobile Computing* 24, 10 (2025), 9289–9302.

[194] Zhixiu Yao, Shichao Xia, Yun Li, and Guangfu Wu. 2023. Cooperative task offloading and service caching for digital twin edge networks: A graph attention multi-agent reinforcement learning approach. *IEEE Journal on Selected Areas in Communications* 41, 11 (2023), 3401–3413.

[195] Mang Ye, Xiuwen Fang, Bo Du, Pong C. Yuen, and Dacheng Tao. 2023. Heterogeneous Federated Learning: State-of-the-art and Research Challenges. *Comput. Surveys* 56, 3 (2023).

[196] Fangfang Yin, Qihong Liu, Danpu Liu, Libiao Jin, and Shufeng Li. 2025. Deep-Reinforcement-Learning-Based Resource Allocation for MEC-Assisted Satellite–Terrestrial Integrated Networks. *IEEE Internet of Things Journal* 13, 1 (2025), 1440–1459.

[197] Siyuan Yin, Haifeng Jiang, Chaogang Tang, Shuo Xiao, and Huaming Wu. 2025. MobiTask: A Federated Learning-Based Task Migration Strategy for Mobile Crowdsensing. *IEEE Transactions on Computational Social Systems* (2025), 1–15.

[198] Ziyan Yin, Zhe Wang, Jun Li, Ming Ding, Wen Chen, and Shi Jin. 2023. Decentralized Federated Reinforcement Learning for User-Centric Dynamic TFDD Control. *IEEE Journal of Selected Topics in Signal Processing* 17, 1 (2023), 40–53.

[199] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. 2022. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in neural information processing systems* 35 (2022), 24611–24624.

[200] Hanqing Yu, Supeng Leng, and Fan Wu. 2024. Joint Cooperative Computation Offloading and Trajectory Optimization in Heterogeneous UAV-Swarm-Enabled Aerial Edge Computing Networks. *IEEE Internet of Things Journal* 11, 10 (2024), 17700–17711.

[201] Honggang Yuan, Ting Wang, Min Fu, and Yuanming Shi. 2025. GIRP: Energy-Efficient QoS-Oriented Microservice Resource Provisioning via Multi-Objective Multi-Task Reinforcement Learning. *IEEE Transactions on Mobile Computing* 24, 7 (2025), 5793–5807.

[202] Yuan Yuan, Bin Yang, Wei Su, Haoru Li, Chang Wang, Qi Liu, and Tarik Taleb. 2024. AoI and energy-driven dynamic cache updates for wireless edge networks. *IEEE Internet of Things Journal* 12, 1 (2024), 792–807.

[203] Zeinab Zabihi, Amir Masoud Eftekhari Moghadam, and Mohammad Hossein Rezvani. 2024. Reinforcement learning methods for computation offloading: a systematic review. *Comput. Surveys* 56, 1 (2024), 1–41.

[204] Ming Zeng, Yanbin Zhao, Jing Wang, and Zesong Fei. 2026. Multiple Reconfigurable Intelligent Surfaces Aided V2X Offloading Networks: A Federated Reinforcement Learning-Based Approach. *IEEE Transactions on Vehicular Technology* 75, 1 (2026), 1281–1294.

[205] Tianyu Zeng, Xiaoxi Zhang, Jingpu Duan, Chao Yu, Chuan Wu, and Xu Chen. 2024. An Offline-Transfer-Online Framework for Cloud-Edge Collaborative Distributed Reinforcement Learning. *IEEE Transactions on Parallel and Distributed Systems* 35, 5 (2024), 720–731.

[206] Cui Zhang, Wenjun Zhang, Qiong Wu, Pingyi Fan, Qiang Fan, Jiangzhou Wang, and Khaled B. Letaief. 2025. Distributed Deep Reinforcement Learning-Based Gradient Quantization for Federated Learning Enabled Vehicle Edge Computing. *IEEE Internet of Things Journal* 12, 5 (2025), 4899–4913.

[207] Guanglin Zhang, Sifan Ni, and Ping Zhao. 2021. Learning-based joint optimization of energy delay and privacy in multiple-user edge-cloud collaboration MEC systems. *IEEE Internet of Things Journal* 9, 2 (2021), 1491–1502.

[208] Junxue Zhang, Xiaodian Cheng, Liu Yang, Jinbin Hu, Ximeng Liu, and Kai Chen. 2024. Sok: Fully homomorphic encryption accelerators. *Comput. Surveys* 56, 12 (2024), 1–32.

[209] Lingling Zhang, Yanxiang Jiang, Fu-Chun Zheng, Mehdi Bennis, and Xiaohu You. 2022. Computation offloading and resource allocation in F-RANs: A federated deep reinforcement learning approach. In *Proceedings of the IEEE international conference on communications workshops*. IEEE, 97–102.

[210] Peiying Zhang, Siyi Li, Lizhuang Tan, Neeraj Kumar, Jian Wang, and Kai Liu. 2026. In-situ data scheduling optimization based on rainbow DQN for IIoT. *Future Generation Computer Systems* 174 (2026), 107940.

[211] Xiangyu Zhang, Dave Biagioni, Mengmeng Cai, Peter Graf, and Saifur Rahman. 2020. An edge-cloud integrated solution for buildings demand response using reinforcement learning. *IEEE Transactions on Smart Grid* 12, 1 (2020), 420–431.

[212] Xiaojie Zhang and Saptarshi Debroy. 2023. Resource management in mobile edge computing: A comprehensive survey. *Comput. Surveys* 55, 13s (2023), 1–37.

[213] Yongchao Zhang, Jia Hu, Geyong Min, Xin Chen, and Nektarios Georgalas. 2024. Joint charging scheduling and computation offloading in EV-assisted edge computing: A safe DRL approach. *IEEE Transactions on Mobile Computing* 23, 9 (2024), 8757–8772.

[214] Zhiyang Zhang, Fengli Zhang, Zehui Xiong, Kuan Zhang, and Dajiang Chen. 2024. LsiA3CS: Deep-Reinforcement-Learning-Based Cloud–Edge Collaborative Task Scheduling in Large-Scale IIoT. *IEEE Internet of Things Journal* 11, 13 (2024), 23917–23930.

[215] Ziyang Zhang, Yang Zhao, Huan Li, Changyao Lin, and Jie Liu. 2024. DVFO: Learning-based DVFS for energy-efficient edge-cloud collaborative inference. *IEEE Transactions on Mobile Computing* 23, 10 (2024), 9042–9059.

[216] Joshua Zhao, Saurabh Bagchi, Salman Avestimehr, Kevin Chan, Somali Chaterji, Dimitris Dimitriadis, Jiacheng Li, Ninghui Li, Arash Nourian, and Holger Roth. 2025. The federation strikes back: A survey of federated learning privacy attacks, defenses, applications, and policy landscape. *Comput. Surveys* 57, 9 (2025), 1–37.

[217] Ming Zhao and Mohammad Reza Nakhai. 2024. A Unified Federated Deep Q Learning Caching Scheme for Scalable Collaborative Edge Networks. *IEEE Transactions on Mobile Computing* 23, 12 (2024), 10855–10866.

[218] Jingjing Zheng, Kai Li, Naram Mhaisen, Wei Ni, Eduardo Tovar, and Mohsen Guizani. 2022. Exploring Deep-Reinforcement-Learning-Assisted Federated Learning for Online Resource Allocation in Privacy-Preserving EdgeIoT. *IEEE Internet of Things Journal* 9, 21 (2022), 21099–21110.

[219] Kechen Zheng, Rongwei Luo, Xiaoying Liu, Jiefan Qiu, and Jia Liu. 2024. Distributed DDPG-based resource allocation for age of information minimization in mobile wireless-powered Internet of Things. *IEEE Internet of Things Journal* 11, 17 (2024), 29102–29115.

[220] Huan Zhou, Hao Wang, Zhiwen Yu, Guo Bin, Mingjun Xiao, and Jie Wu. 2024. Federated Distributed Deep Reinforcement Learning for Recommendation-Enabled Edge Caching. *IEEE Transactions on Services Computing* 17, 6 (2024), 3640–3656.

[221] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2020. A comprehensive survey on transfer learning. *Proceedings of the IEEE* 109, 1 (2020), 43–76.

[222] Mohammad Zolghadri, Parvaneh Asghari, Seyed Ebrahim Dashti, and Alireza Hedayati. 2024. Resource allocation in fog–cloud environments: State of the art. *Journal of Network and Computer Applications* 227 (2024), 103891.

[223] Junfeng Zou, Tongbo Hao, Chen Yu, and Hai Jin. 2021. A3C-DO: A Regional Resource Scheduling Framework Based on Deep Reinforcement Learning in Edge Scenario. *IEEE Transactions on Computers* 70, 2 (2021), 228–239.