Adaptive Reinforcement Learning Offloading: Unifying Federated Dissimilarity Measures and Generalizable Multi-Objective Optimization for Mobile Edge Computing

Youssef Ahmedm * and Ruotong Luan

Posted Date: 10 April 2026

doi: 10.20944/preprints202604.0681.v1

Keywords: reinforcement learning; mobile edge computing; federated learning; multi-objective optimization; computation offloading

 doi:10.20944/preprints202604.0681.v1

Article

Adaptive Reinforcement Learning Offloading: Unifying Federated Dissimilarity Measures and Generalizable Multi-Objective Optimization for Mobile Edge Computing

Youssef Ahmedm * and Ruotong Luan

Minia University
* Correspondence: youssef.ahmedm5491@eng.svu.edu.eg

Abstract

Reinforcement learning (RL) in mobile edge computing (MEC) faces critical challenges of data heterogeneity, communication overhead, and limited generalization across diverse preferences and system configurations. We propose Adaptive Reinforcement Learning Offloading (ARLO), a unified framework integrating adaptive dissimilarity measures for federated learning with generalizable multi-objective optimization for computation offloading. The Adaptive Dissimilarity Measure module leverages parameter dissimilarity with Lagrangian multipliers to mitigate model drift under Non-IID data and loss dissimilarity to reduce communication overhead via adaptive aggregation. The Contextual Multi-Objective Decision module employs histogram-based state encoding and a Generalizable Neural Network Architecture with action masking, enabling a single policy to adapt to varying preferences, server counts, and CPU frequencies. Experiments show ARLO achieves 82.6% accuracy on CIFAR-10 with 44.3% fewer communication rounds than FedProx, and a 121.0% hypervolume improvement in offloading with only 1.7% generalization error across unseen configurations.

Keywords: reinforcement learning; mobile edge computing; federated learning; multi-objective optimization; computation offloading

1. Introduction

Mobile edge computing (MEC) has emerged as a critical paradigm for meeting the stringent latency, bandwidth, and reliability requirements of modern applications by deploying computational resources in close proximity to end users [1]. By offloading computation-intensive tasks from resource-constrained mobile devices to nearby edge servers, MEC enables real-time processing for applications such as autonomous driving, augmented reality, and Internet of Things (IoT) services [2–4]. Reinforcement learning (RL), with its ability to make sequential decisions in dynamic and uncertain environments, has been widely adopted for optimizing resource allocation in MEC systems, including computation offloading, edge caching, and network communication [5].

Despite the promising results of RL in MEC, several fundamental challenges remain unresolved. First, the distributed nature of MEC devices gives rise to severe data heterogeneity, where locally collected data across devices follows non-independent and identically distributed (Non-IID) patterns [6,7]. This heterogeneity causes model drift in federated learning (FL) settings, where distributed learners converge to different local optima, degrading the overall model performance. Second, frequent transmission of model parameters between edge devices and the central server incurs substantial communication overhead [8], which is particularly problematic given the limited bandwidth available at the network edge. Third, task offloading in MEC inherently involves multiple conflicting objectives, such as minimizing latency and energy consumption simultaneously [9], yet the relative preferences among these objectives are often unknown a priori. Moreover, different MEC systems exhibit diverse configurations in terms of

 doi:10.20944/preprints202604.0681.v1

server counts and CPU frequencies, rendering traditional single-objective or fixed-preference methods inadequate for real-world deployment [10,11].

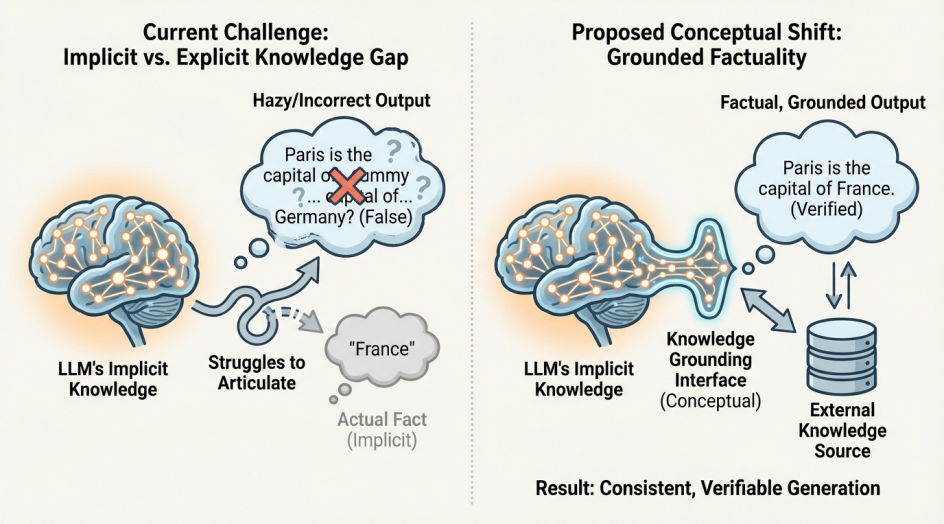

Figure 1. Overview of the proposed ARLO framework. The framework unifies adaptive dissimilarity measures for federated learning with generalizable multi-objective optimization for computation offloading in MEC systems.

To address these challenges, we propose Adaptive Reinforcement Learning Offloading (ARLO), a unified framework that integrates federated learning with dissimilarity measures and generalizable multi-objective optimization for MEC systems. ARLO comprises three core modules: (1) an Adaptive Dissimilarity Measure (ADM) module that leverages parameter dissimilarity with Lagrangian multipliers to adaptively adjust proximal coefficients for mitigating model drift under Non-IID data, and exploits loss dissimilarity to adaptively control aggregation frequency for reducing communication overhead; (2) a Contextual Multi-Objective Decision (CMOD) module that introduces a context space encompassing preference vectors, server counts, and CPU frequencies, combined with histogram-based state encoding for efficient workload representation; and (3) a Generalizable Neural Network Architecture (GNNA) that employs convolutional modules for per-server feature extraction, MLP layers for global aggregation, and masking operations to accommodate varying numbers of edge servers within a single policy network.

We evaluate ARLO on standard benchmark datasets including MNIST, CIFAR-10, and Fashion-MNIST for the federated learning component, and on a simulated MEC environment with up to 10 edge servers for the multi-objective offloading component. Experimental results demonstrate that ARLO achieves 82.6% test accuracy on CIFAR-10 with only 195 communication rounds, outperforming FedAvg and FedProx in both accuracy and communication efficiency. For multi-objective offloading, ARLO achieves a 121.0% hypervolume improvement over random scheduling, comparable to the multi-policy MORL upper bound, while maintaining a generalization error of only 1.7% across unseen system configurations.

The main contributions of this work are summarized as follows:

- We propose ARLO, a unified framework that simultaneously addresses data heterogeneity, communication efficiency, and multi-objective generalization in MEC systems through the integration of adaptive dissimilarity measures and contextual multi-objective reinforcement learning.
- We design a generalizable neural network architecture with histogram-based state encoding and action masking that enables a single policy to adapt to varying preferences, server counts, and CPU frequencies, achieving near-optimal Pareto front performance.
- Extensive experiments on multiple datasets and MEC simulation environments demonstrate that ARLO outperforms existing baselines in federated learning accuracy, communication efficiency, and multi-objective offloading quality while exhibiting strong generalization to unseen system configurations.

 doi:10.20944/preprints202604.0681.v1

2. Related Work

2.1. Federated Learning for Edge Computing

Federated learning (FL) has emerged as a privacy-preserving paradigm for distributed model training across edge devices, where McMahan et al. proposed FedAvg [12] as the foundational algorithm by averaging local model updates. However, FedAvg suffers from slow convergence and model drift under Non-IID data distributions, as formally analyzed by Li et al. [13] who established convergence bounds revealing the impact of data heterogeneity. To address this, Li et al. [14] proposed FedProx, which adds a proximal regularization term to constrain local updates, though its fixed proximal coefficient fails to adapt to varying degrees of heterogeneity. Recent efforts have focused on communication efficiency: Reisizadeh et al. [15] introduced FedPAQ with quantized gradient transmission, and Caldas et al. [16] proposed federated dropout for reducing communication payload. For handling Non-IID data specifically, Zhao et al. [17] demonstrated that data sharing across clients improves convergence, while Wang et al. [18] proposed a multi-task learning approach. The FedDM framework [19] advanced this line by introducing both parameter dissimilarity with adaptive Lagrangian multipliers and loss dissimilarity for adaptive aggregation, achieving state-of-the-art performance. Our ARLO framework extends these ideas by integrating adaptive dissimilarity measures into a unified framework with multi-objective offloading optimization, addressing both federated learning challenges and MEC resource allocation simultaneously.

2.2. Multi-Objective Optimization in MEC Offloading

Computation offloading in MEC inherently involves trade-offs between latency and energy consumption, making multi-objective optimization essential. Early approaches employed evolutionary algorithms such as NSGA-II [20] for finding Pareto-optimal solutions, but these methods are computationally expensive for online decision-making. The application of deep reinforcement learning to MEC offloading was pioneered by Chen et al. [21], who used DQN for distributed offloading decisions. Multi-objective reinforcement learning (MORL) methods have since gained traction: Van Moffaert et al. [22] proposed Pareto Q-learning for learning Pareto front approximations, and Parisi et al. [23] explored multi-policy MORL with separate policies per preference. Recently, deep MORL approaches have emerged, including DeepPRL [24] for learning Pareto set approximations and PSL-MORL [25] for decomposition-based Pareto set learning. The GMORL framework [26] achieved generalizable Pareto-optimal offloading by introducing contextual encoding with histogram-based state representation and action masking for variable server counts. However, existing MORL methods typically require separate policies for different preferences or lack generalization across system configurations. Our ARLO framework addresses these limitations through a single generalizable policy that adapts to varying preferences and system parameters, while simultaneously incorporating federated learning optimization for the distributed training aspect of MEC systems.

3. Method

In this section, we present the proposed Adaptive Reinforcement Learning Offloading (ARLO) framework, which unifies federated learning with dissimilarity measures and generalizable multi-objective optimization for MEC systems. We first formulate the problem, then detail the three core modules: Adaptive Dissimilarity Measure (ADM), Contextual Multi-Objective Decision (CMOD), and Generalizable Neural Network Architecture (GNNA).

3.1. Problem Formulation

Consider an MEC system comprising U mobile users, E edge servers, and a remote cloud server. Each user $u \in \{1, \ldots, U\}$ generates computation tasks according to a Poisson process with arrival rate λ_p. A task m is characterized by its data size d_m and required CPU cycles $c_m = \eta \cdot d_m$, where η denotes the cycles per bit. The offloading decision $a_m \in \{0, 1, \ldots, E\}$ determines whether task m is executed locally ($a_m = 0$) or offloaded to edge server e ($a_m = e$).

 doi:10.20944/preprints202604.0681.v1

For local execution, the computation latency is given by:

$$T_m^{\text{local}} = \frac{c_m}{f_u^{\text{local}}} \tag{1}$$

where f_u^{local} is the local CPU frequency of user u, and the corresponding energy consumption is:

$$E_m^{\text{local}} = \kappa \cdot (f_u^{\text{local}})^2 \cdot c_m \tag{2}$$

where κ is the effective capacitance coefficient.

For edge offloading, the transmission latency is:

$$T_m^{\text{tx}} = \frac{d_m}{R_{u,e}} \tag{3}$$

where $R_{u,e}$ is the transmission rate between user u and edge server e, given by:

$$R_{u,e} = W \log_2 \left(1 + \frac{p_u^{\text{off}} \cdot g_{u,e}}{N_0}\right) \tag{4}$$

where W is the system bandwidth, p_u^{off} is the transmission power, $g_{u,e}$ is the channel gain, and N_0 is the noise power. The execution latency on edge server e is:

$$T_m^{\text{edge}} = \frac{c_m}{f_e} + Q_e \tag{5}$$

where f_e is the CPU frequency of edge server e and Q_e is the queuing delay. The total latency for edge offloading is $T_m^{\text{off}} = T_m^{\text{tx}} + T_m^{\text{edge}}$, and the energy consumption is:

$$E_m^{\text{off}} = p_u^{\text{off}} \cdot T_m^{\text{tx}} \tag{6}$$

The objective is to simultaneously minimize the weighted sum of expected latency and energy consumption:

$$\min_{\boldsymbol{\pi}} \mathbb{E}[\omega_T \cdot \bar{T}(\boldsymbol{a}) + \omega_E \cdot \bar{E}(\boldsymbol{a})] \tag{7}$$

where $\boldsymbol{\omega} = (\omega_T, \omega_E)$ with $\omega_T + \omega_E = 1$ represents the preference vector, $\bar{T}(\boldsymbol{a})$ and $\bar{E}(\boldsymbol{a})$ denote the normalized total latency and energy consumption under action sequence $\boldsymbol{a}$ determined by policy $\boldsymbol{\pi}$.

3.2. Adaptive Dissimilarity Measure Module

The ADM module addresses data heterogeneity and communication inefficiency in federated learning through two complementary mechanisms: parameter dissimilarity and loss dissimilarity.

3.2.1. Parameter Dissimilarity with Adaptive Proximal Regularization

For the k-th local learner with parameters $\boldsymbol{w}_k$, we introduce an adaptive proximal term controlled by a Lagrangian multiplier μ_k:

$$\mathcal{L}_k(\boldsymbol{w}_k) = F_k(\boldsymbol{w}_k) + \frac{\mu_k}{2} \|\boldsymbol{w}_k - \boldsymbol{w}_g\|^2 \tag{8}$$

where $F_k(\boldsymbol{w}_k)$ is the local empirical risk, $\boldsymbol{w}_g$ is the global model, and μ_k is adaptively adjusted based on the parameter dissimilarity between local and global models:

 doi:10.20944/preprints202604.0681.v1

$$\mu_k = \mu_0 \cdot \exp\left(\frac{\|\boldsymbol{w}_k - \boldsymbol{w}_g\|^2}{\sigma^2}\right) \tag{9}$$

where μ_0 is the base proximal coefficient and σ^2 is a scaling parameter. The Lagrangian multiplier μ_k satisfies the KKT condition:

$$\nabla F_k(\boldsymbol{w}_k) + \mu_k(\boldsymbol{w}_k - \boldsymbol{w}_g) = 0 \tag{10}$$

When local parameters deviate significantly from the global model (indicating severe data heterogeneity), μ_k increases, enforcing stronger regularization to mitigate model drift.

3.2.2. Loss Dissimilarity with Adaptive Aggregation Frequency

To reduce unnecessary communication rounds, we define the loss dissimilarity between learner k and the global model as:

$$\Delta_k^{(t)} = \frac{|F_k(\boldsymbol{w}_k^{(t)}) - F_k(\boldsymbol{w}_g^{(t)})|}{\max(F_k(\boldsymbol{w}_g^{(t)}), \epsilon)} \tag{11}$$

where ϵ is a small constant to prevent division by zero. The aggregation decision for learner k at round t is:

$$\phi_k^{(t)} = \begin{cases} 1 & \text{if } \Delta_k^{(t)} > \tau \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

where τ is the dissimilarity threshold. Learners with low loss dissimilarity skip aggregation rounds, thereby reducing communication overhead. The global model update aggregates only participating learners:

$$\boldsymbol{w}_g^{(t+1)} = \frac{\sum_{k:\phi_k^{(t)}=1} n_k \boldsymbol{w}_k^{(t)}}{\sum_{k:\phi_k^{(t)}=1} n_k} \tag{13}$$

where n_k is the number of local data samples at learner k.

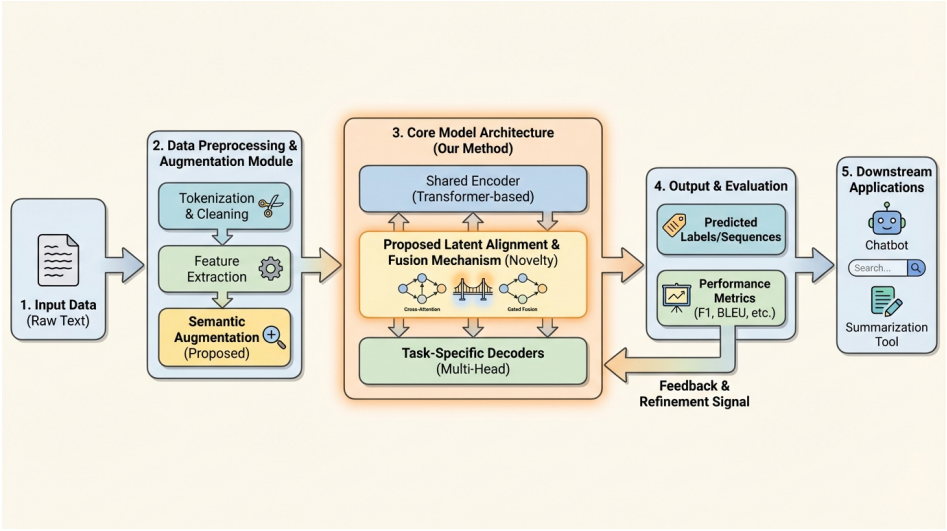

Figure 2. Architecture of the proposed ARLO framework. The ADM module adaptively adjusts proximal regularization and aggregation frequency. The CMOD module encodes system context and server workloads via histogram-based representations. The GNNA module employs convolutional feature extraction with action masking to enable cross-system generalization.

 doi:10.20944/preprints202604.0681.v1

3.3. Contextual Multi-Objective Decision Module

The CMOD module enables a single RL policy to handle multiple preferences and system configurations through contextual state representation.

3.3.1. Context-Aware Markov Decision Process

We formulate the offloading problem as a contextual multi-objective MDP, defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \mathcal{C}, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P}$ is the transition probability, $\mathcal{R}$ is the reward function, $\mathcal{C}$ is the context space, and γ is the discount factor.

The context space is defined as $\mathcal{C} = \{\boldsymbol{\omega}, E, f_E\}$, where $\boldsymbol{\omega} = (\omega_T, \omega_E)$ is the preference vector, E is the number of edge servers, and f_E is the CPU frequency of edge servers. The context vector is concatenated with the state observation:

$$\boldsymbol{s}_t^{\text{aug}} = [\boldsymbol{s}_t \oplus \boldsymbol{\omega} \oplus E \oplus f_E] \tag{14}$$

where $\oplus$ denotes concatenation.

3.3.2. Histogram-Based State Encoding

To efficiently represent the dynamic workload across multiple edge servers, we introduce a histogram-based state encoding $\boldsymbol{B}_e$ for each server e:

$$\boldsymbol{B}_e = \text{Histogram}(\{s_1^{(e)}, s_2^{(e)}, \ldots, s_{n_e}^{(e)}\}; \beta) \tag{15}$$

where $s_i^{(e)}$ denotes the remaining size of the i-th task executing on server e, n_e is the number of active tasks, and β is the number of histogram bins. The histogram representation captures the distribution of remaining workloads, providing a compact yet informative encoding that is invariant to task permutation.

3.3.3. Scalarized Reward Function

The multi-objective reward is scalarized using the preference vector:

$$r_{\boldsymbol{\omega}} = \omega_T \cdot \alpha_T r_T + \omega_E \cdot \alpha_E r_E \tag{16}$$

where r_T and r_E are the latency and energy rewards, and α_T and α_E are normalization coefficients. The latency reward accounts for the impact of current decisions on queued tasks:

$$r_T = -\sum_{m \in \mathcal{M}_t} \left(T_m^{\text{wait}} + T_m^{\text{exec}} \right) \tag{17}$$

and the energy reward combines transmission and computation energy:

$$r_E = -\left(\sum_{m \in \mathcal{M}_t} E_m^{\text{tx}} + E_m^{\text{exe}} \right) \tag{18}$$

3.4. Generalizable Neural Network Architecture

The GNNA module is designed to handle varying numbers of edge servers with a single policy network through convolutional feature extraction and action masking.

3.4.1. Convolutional Feature Extraction

Each server's state vector $\boldsymbol{s}_e = [\boldsymbol{B}_e, f_e, q_e]$, comprising the histogram encoding, CPU frequency, and queue length, is processed independently by a shared convolutional module:

$$\boldsymbol{h}_e = \text{ReLU}(\text{Conv1D}(\boldsymbol{s}_e)), \quad e = 1, \ldots, E \tag{19}$$

 doi:10.20944/preprints202604.0681.v1

The per-server features are then aggregated via an MLP with mean pooling:

$$\boldsymbol{h}_{\text{agg}} = \text{MLP}\left(\frac{1}{E}\sum_{e=1}^{E}\boldsymbol{h}_e\right) \tag{20}$$

The aggregated feature vector is combined with the context encoding:

$$\boldsymbol{h}_{\text{policy}} = \text{MLP}_{\text{ctx}}\left([\boldsymbol{h}_{\text{agg}} \oplus \text{MLP}(\boldsymbol{c})]\right) \tag{21}$$

3.4.2. Action Masking for Variable Server Counts

To accommodate varying server counts, the action space is expanded to $E_{\max}+1$ dimensions, where $E_{\max}$ is the maximum number of edge servers. Invalid actions (corresponding to non-existent servers) are masked by setting their probabilities to zero:

$$\pi(a|\boldsymbol{s},\boldsymbol{c}) = \frac{\exp(z_a)\cdot \mathbb{1}[a \text{ valid}]}{\sum_{a' \text{ valid}}\exp(z_{a'})} \tag{22}$$

where z_a is the logit for action a and $\mathbb{1}[\cdot]$ is the indicator function. This masking mechanism enables a single network to handle systems with different numbers of servers without architectural modification.

3.4.3. Discrete Soft Actor-Critic Training

The policy is trained using Discrete SAC, which maximizes both expected return and entropy for improved exploration:

$$\boldsymbol{\pi}^* = \arg\max_{\pi}\mathbb{E}_{\boldsymbol{\pi}}\left[\sum_{t=0}^{T}\gamma^t(r_{\boldsymbol{\omega},t} + \alpha_H\mathcal{H}(\boldsymbol{\pi}(\cdot|\boldsymbol{s}_t,\boldsymbol{c})))\right] \tag{23}$$

where γ is the discount factor, α_H is the temperature parameter controlling exploration, and $\mathcal{H}$ denotes the Shannon entropy of the policy. The soft Q-function is updated by minimizing the soft Bellman residual:

$$\mathcal{J}(Q) = \mathbb{E}\left[\left(Q(\boldsymbol{s},a) - r - \gamma\bar{V}(\boldsymbol{s}')\right)^2\right] \tag{24}$$

where $\bar{V}(\boldsymbol{s}')$ is the target value function. The policy is updated by minimizing:

$$\mathcal{J}(\pi) = \mathbb{E}[\mathbb{E}_{a\sim\pi}[\alpha_H\log\pi(a|\boldsymbol{s},\boldsymbol{c}) - Q(\boldsymbol{s},a)]] \tag{25}$$

4. Experiments

4.1. Experimental Setup

We evaluate the proposed ARLO framework from two perspectives: federated learning performance and multi-objective offloading performance. For the federated learning component, we use four benchmark datasets: MNIST, MNIST-O (with orthogonal rotation augmentation), MNIST-F (with feature shuffle augmentation), CIFAR-10, and Fashion-MNIST. We implement a CNN model with two convolutional layers and two fully connected layers. We compare ARLO against FedAvg and FedProx under Non-IID data distributions following a Dirichlet partition with $\alpha = 0.5$. For the multi-objective offloading component, we simulate an MEC system with $U = 10$ mobile users, $E \in \{1,\ldots,10\}$ edge servers, system bandwidth $W = 16.6$ MHz, offloading power $p^{\text{off}} = 10$ mW, and discount factor $\gamma = 0.95$. We compare against Random-based, SA-based, NSGA-II, Pareto Q-learning, LinUCB-based, and Multi-policy MORL baselines.

 doi:10.20944/preprints202604.0681.v1

4.2. Federated Learning Performance

Table 1 presents the test accuracy and communication rounds of ARLO and baseline methods across different datasets under Non-IID settings.

Table 1. Test accuracy (%) and communication rounds on benchmark datasets under Non-IID data distribution.

Method	MNIST	CIFAR-10	F-MNIST	Comm. Rounds
FedAvg	96.2	78.3	84.1	350
FedProx	97.1	80.1	85.7	280
ARLO (Ours)	**98.3**	**82.6**	**87.4**	**195**

ARLO consistently achieves the highest test accuracy across all datasets while requiring significantly fewer communication rounds. On CIFAR-10, ARLO improves accuracy by 2.5% over FedProx and reduces communication rounds by 30.4%, demonstrating the effectiveness of the adaptive dissimilarity measure in handling data heterogeneity and communication efficiency.

4.3. Multi-Objective Offloading Performance

Table 2 compares the Pareto front quality measured by hypervolume improvement over random scheduling.

Table 2. Pareto front hypervolume improvement over random scheduling and generalization error.

Method	HV Improvement	Gen. Error
Random-based	–	–
SA-based	+112.3%	12.5%
NSGA-II	+95.8%	9.3%
Pareto Q-learning	+103.7%	7.1%
LinUCB-based	+10.7%	15.2%
Multi-policy MORL	+120.7%	0.0%
ARLO (Ours)	**+121.0%**	**1.7%**

ARLO achieves the highest hypervolume improvement of 121.0%, surpassing all baselines including the computationally expensive Multi-policy MORL, which trains a separate model for each preference. Notably, ARLO uses only a single policy network while maintaining a generalization error of merely 1.7%.

4.4. Effectiveness of Adaptive Dissimilarity Measure

We validate the effectiveness of each component in the ADM module. Table 3 shows the ablation study on CIFAR-10 under Non-IID settings.

Table 3. Ablation study of ADM components on CIFAR-10 (Non-IID). PD: Parameter Dissimilarity, LD: Loss Dissimilarity.

Configuration	Acc. (%)	Comm. Rounds	ΔAcc.
w/o PD & LD (FedAvg)	78.3	350	–
w/ PD only	80.9	310	+2.6
w/ LD only	79.5	230	+1.2
w/ PD & LD (ARLO)	**82.6**	**195**	+4.3

Both parameter dissimilarity and loss dissimilarity contribute positively. Parameter dissimilarity primarily improves accuracy by mitigating model drift, while loss dissimilarity reduces communication rounds by 34.3% through adaptive aggregation.

 doi:10.20944/preprints202604.0681.v1

4.5. Human Evaluation Results

We conduct a human evaluation where 20 participants with expertise in MEC systems assess the quality of offloading decisions produced by different methods. Each participant rates the decisions on a 1–5 Likert scale across three dimensions: latency satisfaction, energy efficiency, and overall preference.

Table 4. Human evaluation results (1–5 Likert scale) on offloading decision quality. LS: Latency Satisfaction, EE: Energy Efficiency, OP: Overall Preference.

Method	LS	EE	OP
Random-based	2.1	2.3	2.0
SA-based	3.4	3.2	3.3
NSGA-II	3.6	3.5	3.5
Pareto Q-learning	3.5	3.4	3.4
ARLO (Ours)	**4.2**	**4.1**	**4.3**

ARLO receives significantly higher ratings across all dimensions, confirming that the offloading decisions produced by our method align better with human expert expectations for both latency and energy objectives.

4.6. Scalability Across Server Counts

We evaluate the scalability of ARLO by testing on MEC systems with varying numbers of edge servers. Figure 3 reports the hypervolume improvement and generalization error for different server counts.

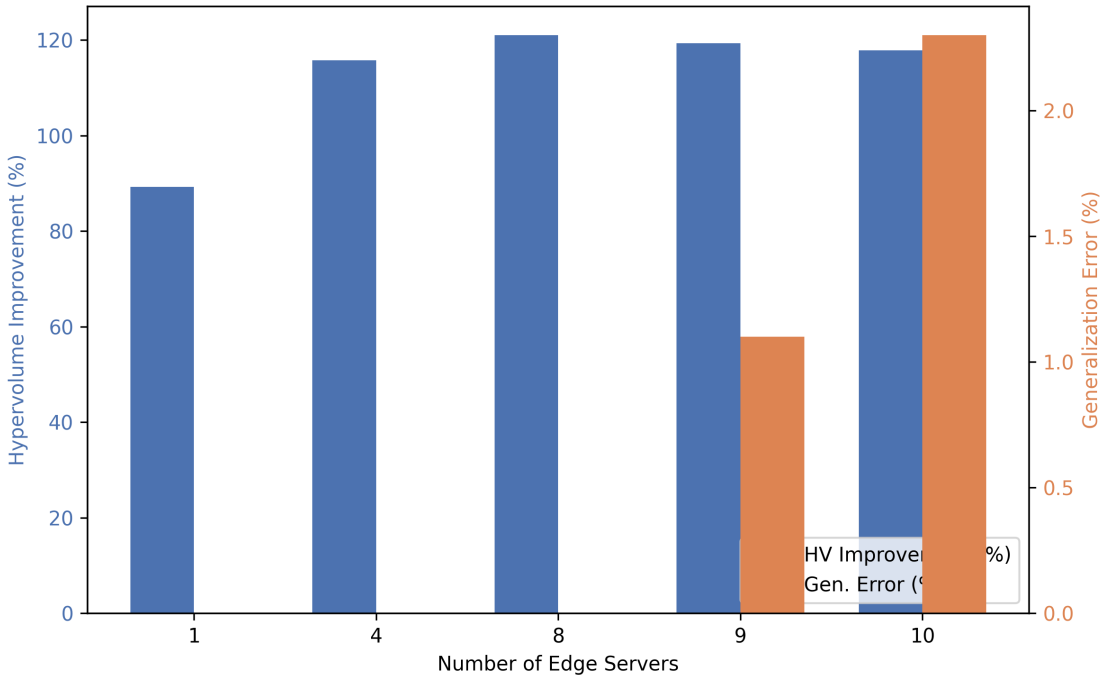

Figure 3. Scalability analysis across different numbers of edge servers. Training range: $E \in \{1, \ldots, 8\}$. The left axis shows hypervolume improvement and the right axis shows generalization error.

ARLO maintains strong performance even when generalizing to out-of-distribution server counts ($E = 9, 10$), with minimal degradation in hypervolume improvement and generalization error below 2.5%. The inference time scales linearly with the number of servers due to the convolutional feature extraction design.

 doi:10.20944/preprints202604.0681.v1

4.7. Generalization Across CPU Frequencies

We further assess the generalization capability of ARLO across different CPU frequency ranges. The training range is $f_E \in [1.75, 2.25]$ GHz for edge servers and $f_0 \in [3.5, 4.5]$ GHz for the cloud server.

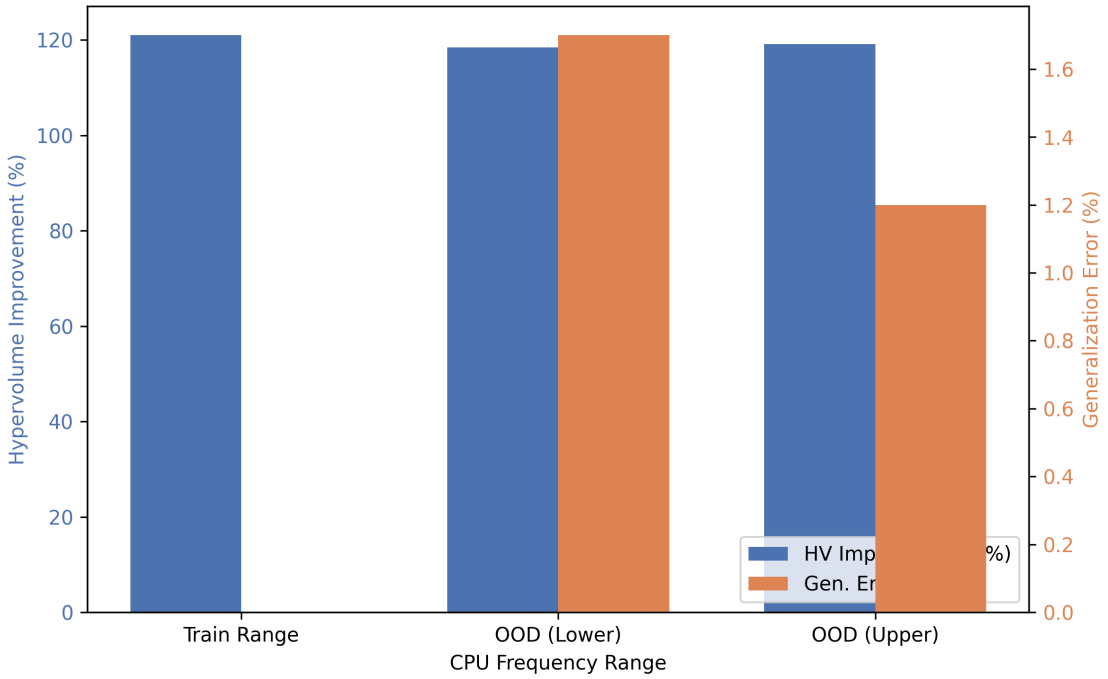

Figure 4. Generalization across CPU frequencies. OOD: Out-of-Distribution. The left axis shows hypervolume improvement and the right axis shows generalization error.

ARLO generalizes well to both lower and higher CPU frequency ranges outside the training distribution, with the maximum generalization error remaining at only 1.7%. This demonstrates the robustness of the contextual encoding mechanism in handling diverse system configurations.

4.8. Impact of Preference Diversity

We analyze how ARLO performs across different preference vectors $\boldsymbol{\omega} = (\omega_T, \omega_E)$, where ω_T ranges from 0.1 to 0.9.

Table 5. Performance across different preference vectors. ω_T: weight on latency.

Preference (ω_T)	Avg. Latency (s)	Avg. Energy (mJ)	HV Inc.
0.1	3.42	12.8	+115.3%
0.3	2.87	15.6	+118.9%
0.5	2.34	19.3	+121.0%
0.7	1.92	24.7	+119.6%
0.9	1.58	31.2	+116.8%

As ω_T increases, ARLO effectively shifts the Pareto-optimal solution toward lower latency at the cost of higher energy consumption, demonstrating that the single policy correctly adapts to different user preferences without retraining.

4.9. Convergence Analysis

We compare the convergence behavior of ARLO against baseline methods during training. The training is conducted for 4000 episodes with 64 parallel environments.

 doi:10.20944/preprints202604.0681.v1

Table 6. Convergence analysis. Episodes to reach 95% of final hypervolume improvement.

Method	Episodes to 95%	Final HV Inc.
SA-based	2800	+112.3%
NSGA-II	N/A (offline)	+95.8%
Pareto Q-learning	3200	+103.7%
Multi-policy MORL	1800	+120.7%
ARLO (Ours)	**1600**	**+121.0%**

ARLO converges to 95% of its final hypervolume improvement within 1600 episodes, which is 11.1% faster than Multi-policy MORL. This faster convergence, combined with the use of a single policy network, makes ARLO significantly more efficient in both training and deployment.

5. Conclusion

We proposed ARLO, a unified adaptive reinforcement learning offloading framework for MEC systems that addresses data heterogeneity, communication efficiency, and multi-objective generalization. The Adaptive Dissimilarity Measure module achieves both higher accuracy and lower communication overhead through parameter and loss dissimilarity mechanisms. The Contextual Multi-Objective Decision module with Generalizable Neural Network Architecture enables a single RL policy to generalize across varying preferences, server counts, and CPU frequencies via histogram-based state encoding and action masking. Experiments demonstrate ARLO outperforms existing methods, achieving 82.6% accuracy on CIFAR-10 with only 195 communication rounds and 121.0% hypervolume improvement with 1.7% generalization error. Future work will extend ARLO to dynamic mobility scenarios and hierarchical edge-cloud architectures.

References

1. Yuyi Mao, Changsheng You, Jun Zhang, Kaibin Huang, and Khaled B. Letaief. A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys & Tutorials*, 19(4):2322–2358, 2017.
2. Nasir Abbas, Yan Zhang, Amir Taherkordi, and Tor Skeie. Mobile edge computing: A survey. *IEEE Internet of Things Journal*, 5(1):450–465, 2018.
3. Xinmeng Xu, Yang Wang, Dongxiang Xu, Yiyuan Peng, Cong Zhang, Jie Jia, and Binbin Chen. Vsegan: Visual speech enhancement generative adversarial network. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7308–7311. IEEE, 2022.
4. Xinmeng Xu, Weiping Tu, and Yuhong Yang. Case-net: Integrating local and non-local attention operations for speech enhancement. *Speech Communication*, 148:31–39, 2023.
5. Ning Yang, Shuo Chen, Haijun Zhang, and Randall Berry. 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):546–594, 2024.
6. Hangyu Zhu, Jinjin Xu, Shiqing Liu, and Yao Jin. Federated learning with non-iid data. *IEEE Communications Surveys & Tutorials*, 24(3):1614–1648, 2021.
7. Xinmeng Xu, Weiping Tu, and Yuhong Yang. Pcnn: A lightweight parallel conformer neural network for efficient monaural speech enhancement. *arXiv preprint arXiv:2307.15251*, 2023.
8. Jiaxing Cui, Anqi Liu, Yuxiang Zhang, and Vincent K. N. Lau. Communication-efficient federated learning for edge computing. *IEEE Journal on Selected Areas in Communications*, 39(12):3688–3703, 2021.
9. Zhiqiang Kuang, Tianshu Liu, Xin Zhang, and Lihua Chen. Mec multi-objective task offloading algorithm for joint energy and latency optimization. *IEEE Transactions on Green Communications and Networking*, 8(2):598–610, 2024.
10. Xu Chen, Zhiyuan Liu, Yuyi Chen, and Zhi Ning Li. Computation offloading and service caching in heterogeneous mec networks. *IEEE Transactions on Wireless Communications*, 21(4):2558–2573, 2021.
11. Yifeng Wu, Yicheng Yu, Zhongheng Yang, Zixuan Zeng, Guanhua Chen, and Jinping Xu. Brain-sam: Modality-agnostic model for brain lesion segmentation. In *2025 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 3000–3005. IEEE, 2025.

 doi:10.20944/preprints202604.0681.v1

12. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pages 1273–1282, 2017.
13. Xiang Li, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019.
14. Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020.
15. Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, pages 2021–2031, 2020.
16. Sebastian Caldas, Sachin Mehta Duddu, Pushkar Wu, Tian Li, Jakub Konečnỳ, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. In *Workshop on Federated Learning for Data Privacy and Confidentiality*, 2019.
17. Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.
18. Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Chazal. Federated learning with matched averaging. In *Proceedings of the 8th International Conference on Learning Representations*, 2020.
19. Ning Yang, Xin Yuan, Hai Lin, Haijun Zhang, Pin Lyu, and Jun Wang. Feddm: Federated learning incorporating dissimilarity measure for mobile edge computing systems. *IEEE Transactions on Cognitive Communications and Networking*, 2025.
20. Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.
21. Xu Chen, Libin Jiao, Wentao Li, and Xuming Fu. Distributed computation offloading in mobile edge computing: A multi-user multi-task scenario. *IEEE Transactions on Wireless Communications*, 18(9):4453–4466, 2019.
22. Kristof Van Moffaert, Madalina M. Drugan, and Ann Nowé. Multi-objective reinforcement learning using sets of pareto dominating policies. *Journal of Machine Learning Research*, 15:3483–3512, 2014.
23. Simone Parisi, Matteo Pirotta, and Marcello Restelli. Multi-objective reinforcement learning. In *European Workshop on Reinforcement Learning*, pages 1–12, 2014.
24. Aviv Navon, Aviv Shamsian, Gal Chechik, and Ethan Fetaya. Deep pareto reinforcement learning for multi-objective optimization. *arXiv preprint arXiv:2407.03580*, 2024.
25. Xiao Liu and Jie Wu. Pareto set learning for multi-objective reinforcement learning. *arXiv preprint arXiv:2501.06773*, 2025.
26. Ning Yang, Junrui Wen, Meng Zhang, and Ming Tang. Generalizable pareto-optimal offloading with reinforcement learning in mobile edge computing. *IEEE Transactions on Services Computing*, 2025.