ADAPT-ROMA: Adaptive Risk-Aware Multi-Objective Offloading for Robust Mobile Edge Computing

Zeren Gu * and Jialei Tan

Posted Date: 8 April 2026

doi: 10.20944/preprints202604.0537.v1

Keywords: mobile edge computing; offloading; risk-aware; multi-objective; deep reinforcement learning

 doi:10.20944/preprints202604.0537.v1

Article

ADAPT-ROMA: Adaptive Risk-Aware Multi-Objective Offloading for Robust Mobile Edge Computing

Zeren Gu * and Jialei Tan

Henan University of Economics and Law, China
* Correspondence: 20236903924@stu.huel.edu.cn

Abstract

Mobile Edge Computing (MEC) offers significant advantages in enhancing user experience and reducing energy consumption by offloading tasks to proximate edge servers. However, MEC systems are inherently complex, facing highly dynamic environments with fluctuating channel conditions and server loads, coupled with the necessity to optimize multiple conflicting objectives such as latency and energy consumption. Furthermore, existing offloading strategies often lack robustness against inherent uncertainties, leading to performance degradation in unpredictable scenarios. To address these critical challenges, this paper introduces ADAPT-ROMA, a novel Adaptive Dynamic Risk-aware Multi-objective Offloading Algorithm. We formulate the MEC offloading problem as a Contextual Risk-aware Multi-objective Markov Decision Process, integrating Conditional Value at Risk (CVaR) to explicitly quantify and mitigate worst-case performance losses. ADAPT-ROMA employs an adaptive weight generation network to dynamically balance latency, energy, and risk objectives based on real-time system states. Its core is built upon a robust distributed deep reinforcement learning framework, enhanced by adversarial training to bolster resilience against unmodeled uncertainties, and a hierarchical attention network for effective contextual encoding. Extensive simulations demonstrate that ADAPT-ROMA achieves a superior balance between average task latency and total energy consumption, alongside an outstanding task completion rate. Crucially, it yields a significantly low risk-aware score, outperforming baseline methods by ensuring high performance and stability even under dynamic and uncertain MEC conditions. Ablation studies, adaptability analysis, robustness evaluation, and scalability tests further confirm the individual contributions of ADAPT-ROMA's components and its practical applicability.

Keywords: mobile edge computing; offloading; risk-aware; multi-objective; deep reinforcement learning

1. Introduction

Mobile Edge Computing (MEC) has emerged as a pivotal paradigm in modern wireless communication systems, significantly enhancing the capabilities of mobile devices by offloading computationally intensive tasks to proximate edge servers [1]. This paradigm promises substantial improvements in user experience through reduced latency, lower energy consumption for terminal devices, and increased computational throughput, which are critical for supporting delay-sensitive applications such as augmented reality, autonomous driving, and real-time industrial IoT [2]. The strategic placement of computing resources closer to data sources at the network edge circumvents the limitations of traditional cloud computing, making MEC an indispensable component of 5G and future 6G networks, with ongoing research exploring various optimization strategies for resource allocation and task offloading [3,4].

Despite its immense potential, MEC offloading systems are fraught with significant challenges that impede their optimal performance. Firstly, the dynamic nature of MEC environments is a major hurdle, encompassing rapidly fluctuating wireless channel conditions, varying loads on edge servers,

 doi:10.20944/preprints202604.0537.v1

and the heterogeneous, stochastic arrival patterns of user tasks [5]. These dynamics necessitate highly adaptive offloading decisions. Secondly, offloading decisions typically involve optimizing multiple conflicting objectives, primarily minimizing task execution latency and reducing energy consumption. Achieving an optimal trade-off between these objectives is a complex multi-objective optimization problem, often tackled using Pareto-optimal approaches to understand inherent trade-offs [4,6]. Lastly, conventional offloading strategies often exhibit fragility when confronted with uncertainties, such as unpredictable future channel states, sudden server failures, or unexpected task surges. This lack of robustness can lead to severe performance degradation in real-world deployments [7]. Addressing such uncertainties often involves developing robust and distributed learning paradigms, like federated learning [8]. Current research often addresses either dynamics or multi-objectives, or robustness in isolation, leaving a critical research gap in developing strategies that simultaneously tackle dynamic uncertainty and multi-objective robustness.

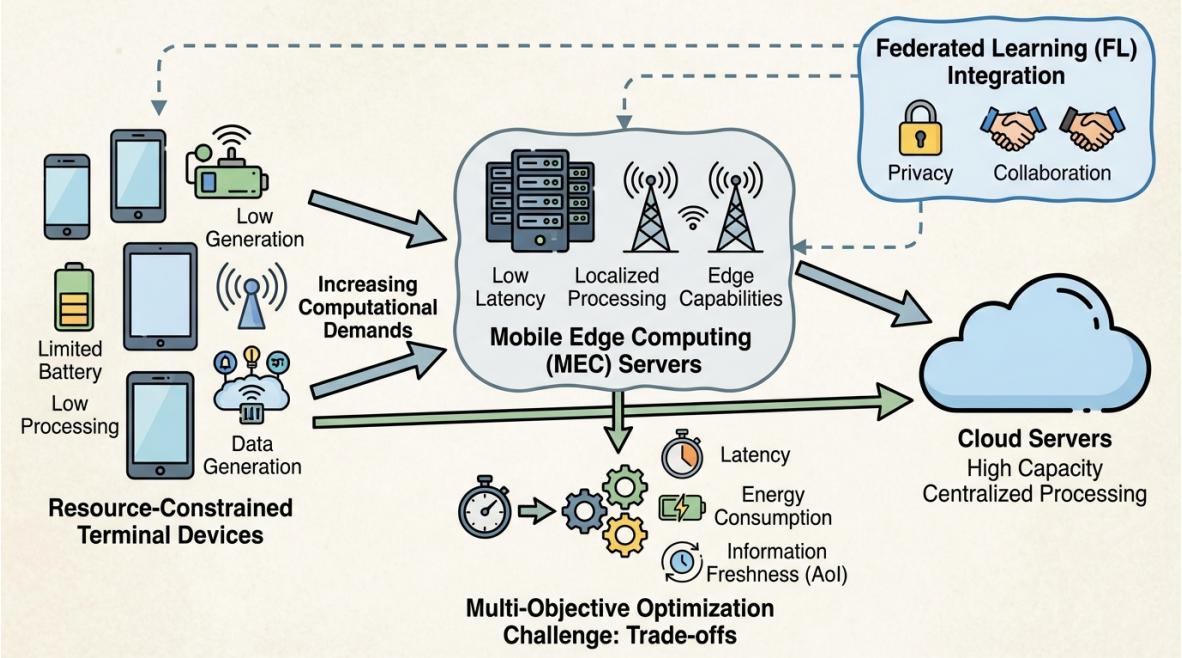

Figure 1. An illustrative overview of the Mobile Edge Computing (MEC) ecosystem. Resource-constrained terminal devices offload computationally intensive tasks to proximate MEC servers to address increasing computational demands. The MEC paradigm targets improvements in user experience by optimizing multiple conflicting objectives, primarily reducing task execution latency and energy consumption. The figure also shows the role of cloud servers and potential integration with Federated Learning.

Motivated by these pressing challenges, this research proposes an adaptive risk-aware multi-objective MEC offloading strategy. Our primary objective is not merely to balance latency and energy consumption in dynamic environments but, more importantly, to endow offloading decisions with a "risk perception" mechanism against future uncertainties. This approach ensures that the offloading strategy maintains high performance and stability even under adverse or highly unpredictable MEC conditions, thereby filling the aforementioned research gap and providing a more resilient and efficient solution.

To this end, we introduce the Adaptive Dynamic Risk-aware Multi-objective Offloading Algorithm (ADAPT-ROMA). Our method first models the MEC offloading problem as a Contextual Risk-aware Multi-objective Markov Decision Process (MOMDP), where the state space comprehensively captures real-time channel conditions, edge server loads, task queue information, and probabilistic estimates of future environmental uncertainties. The action space includes choices such as local execution, offloading to various edge servers, or migrating to the cloud. A novel aspect of ADAPT-ROMA is its risk perception modeling, where we incorporate Conditional Value at Risk (CVaR) as an additional

 doi:10.20944/preprints202604.0537.v1

optimization objective or constraint. CVaR quantifies the expected performance loss in the worst-case scenarios, thereby guiding the policy to avoid extreme adverse outcomes while optimizing average performance. We develop a neural network-based risk prediction module that estimates the worst-case scenarios and their probabilities for different offloading decisions in real time. The core of ADAPT-ROMA is built upon a sophisticated multi-objective deep reinforcement learning (DRL) framework. This framework features a novel adaptive weight generation network that dynamically adjusts the weights among latency, energy consumption, and risk (CVaR) objectives based on current system states and user preferences, transforming the multi-objective problem into a dynamically weighted single-objective optimization. Furthermore, a distributed DRL (Distributed DRL) architecture, coupled with adversarial training techniques (e.g., introducing adversarial perturbations to simulate uncertain environments), enhances the strategy's robustness against unknown channel fading or task bursts, drawing inspiration from distributed and robust learning frameworks in related fields [8]. A hierarchical attention network acts as a contextual encoder, processing high-dimensional and heterogeneous contextual information (e.g., server CPU frequency, bandwidth, user location, task type) to ensure the strategy's generalization capability across diverse MEC deployments and system parameters, a technique widely recognized for capturing complex dependencies in deep learning architectures [9].

To validate the efficacy of ADAPT-ROMA, we conduct extensive simulations within a custom-built MEC environment. Our simulation setup, developed in Python, accurately models a realistic MEC system comprising multiple mobile user devices, several edge servers, and a cloud server. The environment simulates 15 mobile users, each generating heterogeneous computing tasks according to a Poisson process, varying in computational load and data size. Wireless channel uncertainty is modeled using a time-varying Rayleigh fading channel. We deploy 3 edge servers, each possessing distinct computational capabilities and communication bandwidths, whose load states are updated in real-time. For performance evaluation, we benchmark ADAPT-ROMA against several established and baseline offloading strategies: a Greedy Offloading approach (prioritizing the least loaded server or local execution), Delay-first DRL (a DRL strategy solely optimizing for minimal latency), Energy-first DRL (a DRL strategy solely optimizing for minimal energy consumption), and a Basic Multi-objective DRL (MORL) method employing static objective weights.

The experimental results (as detailed in Table 1 in the main body) demonstrate ADAPT-ROMA's superior comprehensive performance across critical metrics. While ADAPT-ROMA achieves an average task latency of 48.2 ms and total energy consumption of 118.5 mJ/cycle, exhibiting a well-balanced trade-off compared to single-objective DRL baselines, its most notable advantage lies in its task completion rate of 95.1% and an outstandingly low risk-aware score of 0.15. This significantly outperforms all comparison methods, indicating ADAPT-ROMA's unparalleled ability to process dynamic tasks efficiently and, crucially, to mitigate the impact of worst-case scenarios under uncertainty. The low risk-aware score confirms the strategy's robustness, ensuring high performance even when faced with unpredictable channel degradation or server overloads. Compared to the Basic Multi-objective DRL, ADAPT-ROMA's adaptive weighting and risk perception module yield improvements across all metrics, underscoring the benefits of dynamic adjustments and explicit risk quantification in multi-objective optimization.

In summary, this paper makes the following key contributions:

- We formulate a novel Contextual Risk-aware Multi-objective Markov Decision Process (MOMDP) for MEC offloading, explicitly incorporating environmental uncertainties and multiple conflicting objectives.
- We propose ADAPT-ROMA, a sophisticated deep reinforcement learning algorithm featuring a CVaR-based risk prediction module, an adaptive weight generation network for dynamic multi-objective optimization, and a robust distributed DRL framework with contextual encoding.

 doi:10.20944/preprints202604.0537.v1

- We rigorously evaluate ADAPT-ROMA in a simulated dynamic MEC environment, demonstrating its superior performance in terms of balanced latency and energy, high task completion rate, and significantly enhanced robustness against uncertainties as quantified by its low risk-aware score.

2. Related Work

2.1. Deep Reinforcement Learning for MEC Offloading

Mobile Edge Computing (MEC) is pivotal for meeting low-latency, high-computation demands at the network edge, primarily through offloading tasks from mobile devices to edge servers. This complex decision-making process in dynamic, multi-user environments benefits significantly from Deep Reinforcement Learning (DRL) due to its ability to manage sequential decisions within vast state spaces. Foundational research focuses on MEC architecture for efficient offloading, optimizing metrics such as Age of Information (AoI), developing generalizable solutions, and exploring Federated Learning within MEC [3,4,8,10]. DRL-based task and computation offloading mechanisms are crucial for optimizing resource utilization, latency, and throughput [11,12], including generalizable and Pareto-optimal policies [4]. DRL provides an adaptive paradigm for real-time offloading decisions in intricate edge scenarios [13,14]. Advanced DRL applications encompass dynamic offloading strategies that adapt to network conditions and user mobility [5], multi-user scenarios managing simultaneous requests and resource allocation [15], and energy efficiency optimization [16].

2.2. Multi-Objective and Risk-Aware Reinforcement Learning

Reinforcement learning (RL) increasingly addresses complex, uncertain environments through Multi-Objective Reinforcement Learning (MORL) and Risk-Aware Reinforcement Learning. MORL explores methods for agents to simultaneously optimize multiple, often conflicting, reward functions [17], including generalizable Pareto-optimal offloading policies [4]. Pareto optimization offers a framework for identifying non-dominated solutions and understanding trade-offs [4,18]. Scalarization techniques are widely used to convert multiple objectives into a single weighted function for standard RL algorithms [19], with dynamic weight adjustment further enabling adaptive objective prioritization [20]. Risk-aware RL explicitly incorporates risk metrics to ensure policy robustness and safety against adverse outcomes, considering factors like Age of Information in dynamic systems or robustness challenges in federated learning [3,8]. Conditional Value at Risk (CVaR) is a crucial metric, quantifying expected loss in worst-case outcomes for robust downside risk control [21]. Effective risk management also necessitates robust uncertainty quantification to model environmental unpredictability and inform cautious policy design [1]. Adversarial training enhances policy resilience by exposing agents to challenging scenarios, improving performance in worst-case conditions [22,23]. Advanced deep learning architectures, such as those with attention mechanisms, contribute to more robust and generalizable learning systems [9,24]. Ultimately, the principle of robust control guides the design of policies that maintain performance guarantees under various uncertainties, fostering reliable and safe risk-aware RL systems [25].

3. Method

This section details the proposed **Adaptive Dynamic Risk-aware Multi-objective Offloading Algorithm (ADAPT-ROMA)**, which is designed to address the challenges of dynamic environments, conflicting objectives, and inherent uncertainties in Mobile Edge Computing (MEC) offloading. We first outline the system model and formally formulate the problem, including the communication and computation costs. Subsequently, we describe our novel risk perception modeling using Conditional Value at Risk (CVaR) and then present the core components of the ADAPT-ROMA framework.

3.1. System Model and Problem Formulation

We consider a typical MEC system comprising a set of U mobile users, $\mathcal{U} = \{u_1, u_2, \ldots, u_U\}$, a set of M edge servers, $\mathcal{E} = \{e_1, e_2, \ldots, e_M\}$, and a remote cloud server e_{M+1}. Each mobile user $u \in \mathcal{U}$

 doi:10.20944/preprints202604.0537.v1

generates computational tasks in discrete time slots $t \in \{1, 2, \ldots, T\}$. Each task $k_u(t)$ is characterized by its input data size $D_u(t)$ (in bits) and required CPU cycles $C_u(t)$ (in cycles).

3.1.1. Communication and Computation Model

For a mobile user u to offload a task $k_u(t)$ to an edge server $e \in \mathcal{E}$ or the cloud e_{M+1}, data transmission is required. The wireless channel condition between user u and server e at time t is denoted by $h_{u,e}(t)$, and the allocated bandwidth is $B_{u,e}(t)$. Assuming a fixed transmit power $P_u(t)$ for user u and background noise power N_0, the data transmission rate $R_{u,e}(t)$ can be modeled using Shannon's formula:

$$R_{u,e}(t) = B_{u,e}(t) \log_2 \left(1 + \frac{P_u(t) h_{u,e}(t)}{N_0}\right) \tag{1}$$

The transmission latency $L^{\text{trans}}_{u,e}(t)$ and transmission energy consumption $E^{\text{trans}}_{u,e}(t)$ for offloading task $k_u(t)$ to server e are:

$$L^{\text{trans}}_{u,e}(t) = \frac{D_u(t)}{R_{u,e}(t)} \tag{2}$$

$$E^{\text{trans}}_{u,e}(t) = P_u(t) \cdot L^{\text{trans}}_{u,e}(t) \tag{3}$$

Upon successful transmission, the task is executed at the chosen server. Each server e (including local device and cloud) has a computational capacity represented by its CPU frequency F_e (in cycles/second). The execution latency $L^{\text{exec}}_{u,e}(t)$ for task $k_u(t)$ on server e is:

$$L^{\text{exec}}_{u,e}(t) = \frac{C_u(t)}{F_e} \tag{4}$$

For local execution, the user's device CPU frequency is F^{local}_u, and the corresponding execution energy $E^{\text{local}}_u(t)$ is modeled as:

$$E^{\text{local}}_u(t) = \kappa (F^{\text{local}}_u)^3 C_u(t) \tag{5}$$

where κ is an effective capacitance coefficient related to the chip architecture. Energy consumed by edge/cloud servers is assumed to be handled by the server provider and not explicitly considered in the user's energy objective.

The total latency $L_u(t)$ for user u's task $k_u(t)$ depends on the offloading decision $a_u(t)$:

$$L_u(t) = \begin{cases} L^{\text{local}}_u(t) & \text{if } a_u(t) = \text{Local Execution} \\ L^{\text{trans}}_{u,a_u(t)}(t) + L^{\text{exec}}_{u,a_u(t)}(t) & \text{if } a_u(t) \in \mathcal{E} \cup \{e_{M+1}\} \end{cases} \tag{6}$$

Similarly, the total energy consumption $E_u(t)$ for user u's task $k_u(t)$ is:

$$E_u(t) = \begin{cases} E^{\text{local}}_u(t) & \text{if } a_u(t) = \text{Local Execution} \\ E^{\text{trans}}_{u,a_u(t)}(t) & \text{if } a_u(t) \in \mathcal{E} \cup \{e_{M+1}\} \end{cases} \tag{7}$$

3.1.2. Contextual Risk-aware Multi-objective Markov Decision Process

The MEC offloading problem is modeled as a **Contextual Risk-aware Multi-objective Markov Decision Process (MOMDP)**. The state space $\mathcal{S}$ at time t comprehensively captures the real-time environmental context and perceived uncertainties:

$$\mathbf{s}(t) = \{\mathbf{h}(t), \mathbf{L}(t), \mathbf{Q}(t), \boldsymbol{\sigma}(t), \mathbf{x}(t)\} \tag{8}$$

 doi:10.20944/preprints202604.0537.v1

where $\mathbf{h}(t)$ represents the vector of wireless channel gains $h_{u,e}(t)$ between users and potential offloading destinations; $\mathbf{L}(t)$ denotes the current computational load and available CPU frequencies F_e of all edge servers; $\mathbf{Q}(t)$ includes information on task queues (e.g., task counts, remaining deadlines) for local execution and on each edge server; $\sigma(t)$ is a probabilistic estimation of future environmental uncertainties, such as predicted variance in channel quality or server load fluctuations, derived from historical data or predictive models; and $\mathbf{x}(t)$ encapsulates other static or slowly changing contextual information (e.g., user locations, task types, $P_u(t)$, $B_{u,e}(t)$, N_0, κ).

The action space $\mathcal{A}$ for each user u at time t consists of choosing an offloading destination:

$$a_u(t) \in \{\text{Local Execution}, e_1, e_2, \ldots, e_M, e_{M+1}\} \tag{9}$$

A joint action for all users is denoted as $\mathbf{a}(t) = \{a_1(t), \ldots, a_U(t)\}$. The system transitions from state $\mathbf{s}(t)$ to $\mathbf{s}(t+1)$ based on $\mathbf{a}(t)$ and stochastic environmental dynamics.

The primary objectives are to minimize total task execution latency, total energy consumption, and the risk of extreme performance degradation. For a given state-action pair $(\mathbf{s}(t), \mathbf{a}(t))$, we define the immediate cost vector $\mathbf{c}(\mathbf{s}(t), \mathbf{a}(t)) = [\mathcal{L}(\mathbf{s}(t), \mathbf{a}(t)), \mathcal{E}(\mathbf{s}(t), \mathbf{a}(t)), \mathcal{R}(\mathbf{s}(t), \mathbf{a}(t))]$, where:

$$\mathcal{L}(\mathbf{s}(t), \mathbf{a}(t)) = \sum_{u \in \mathcal{U}} L_u(t) \quad \text{(from Equation (6))} \tag{10}$$

$$\mathcal{E}(\mathbf{s}(t), \mathbf{a}(t)) = \sum_{u \in \mathcal{U}} E_u(t) \quad \text{(from Equation (7))} \tag{11}$$

The risk metric $\mathcal{R}(\mathbf{s}(t), \mathbf{a}(t))$ is derived from our CVaR modeling, as elaborated in Section 3.2. The overall objective is to find an optimal offloading policy $\pi : \mathcal{S} \to \mathcal{A}$ that minimizes a dynamically weighted sum of these objectives over a long horizon, subject to operational constraints. Key constraints include:

$$L_u(t) \leq D_u^{\max}(t) \qquad \forall u \in \mathcal{U}, t \in T \tag{12}$$

$$\sum_{u \in \mathcal{U}, a_u(t) = e} C_u(t)/F_e \leq T_{\text{slot}} \qquad \forall e \in \mathcal{E} \cup \{e_{M+1}\}, t \in T \tag{13}$$

$$E_u(t) \leq B_u^{\text{rem}}(t) \qquad \forall u \in \mathcal{U}, t \in T \tag{14}$$

where $D_u^{\max}(t)$ is the deadline for task $k_u(t)$, T_{slot} is the duration of a time slot, and $B_u^{\text{rem}}(t)$ is the remaining battery capacity of user u. The main objective is:

$$\min_{\pi} \mathbb{E}_{\pi} \left[\sum_{t=0}^{T} (w_L(t) \cdot \mathcal{L}(\mathbf{s}(t), \mathbf{a}(t)) + w_E(t) \cdot \mathcal{E}(\mathbf{s}(t), \mathbf{a}(t)) + w_{CVaR}(t) \cdot \mathcal{R}(\mathbf{s}(t), \mathbf{a}(t))) \right] \tag{15}$$

where $w_L(t)$, $w_E(t)$, and $w_{CVaR}(t)$ are dynamically generated weights for latency, energy, and risk, respectively, such that $w_L(t) + w_E(t) + w_{CVaR}(t) = 1$. The expectation is taken over the stochastic transitions and chosen actions.

3.2. Risk-Aware Modeling with CVaR

To address the fragility of traditional strategies against uncertainties, we introduce a rigorous **risk perception mechanism** into the offloading decision-making process. We model risk using **Conditional Value at Risk (CVaR)**, which quantifies the expected loss in the worst α-percentile of outcomes. Unlike Value at Risk (VaR), which only indicates the threshold loss, CVaR provides insight into the magnitude of loss beyond that threshold, making it a robust measure for risk management.

Let X be a random variable representing a critical performance cost (e.g., a combined metric of latency and energy) incurred by an offloading decision under uncertain future conditions. For a given

 doi:10.20944/preprints202604.0537.v1

confidence level $\alpha \in (0,1)$, the VaR of X at level α, denoted $VaR_\alpha(X)$, is defined as the minimum cost value such that the probability of the cost not exceeding this value is at least α:

$$VaR_\alpha(X) = \inf\{x \in \mathbb{R} : P(X \leq x) \geq \alpha\} \tag{16}$$

The CVaR of X at level α, denoted $CVaR_\alpha(X)$, is the expected cost given that the cost exceeds $VaR_\alpha(X)$:

$$CVaR_\alpha(X) = \mathbb{E}[X | X \geq VaR_\alpha(X)] \tag{17}$$

An equivalent and often more tractable formulation for optimization purposes, which we leverage, is given by:

$$CVaR_\alpha(X) = \min_{\nu \in \mathbb{R}} \left(\nu + \frac{1}{1-\alpha} \mathbb{E}[\max(0, X - \nu)] \right) \tag{18}$$

where ν is an auxiliary variable representing a potential VaR estimate, and the expectation is taken over the distribution of X.

In ADAPT-ROMA, we treat CVaR as an explicit objective to be minimized, denoted as $\mathcal{R}(\mathbf{s}(t), \mathbf{a}(t))$ in Equation (15). We develop a **neural network-based risk prediction module** that takes the current state $\mathbf{s}(t)$ and a candidate action $\mathbf{a}(t)$ as input. This module is trained to estimate the distribution of future costs (specifically, a combined latency and energy metric, say $X_u(t) = \lambda L_u(t) + \epsilon E_u(t)$ for each user or aggregated) under various uncertain scenarios (e.g., rapid channel degradation, unexpected server load surges). Specifically, for each state-action pair $(\mathbf{s}(t), \mathbf{a}(t))$, the neural network outputs parameters (e.g., mean and variance if assuming a Gaussian distribution, or quantiles for a more general distribution) of the predicted cost distribution $P(X|\mathbf{s}(t), \mathbf{a}(t))$. From this predicted distribution, we can then numerically compute $CVaR_\alpha(X(\mathbf{s}(t), \mathbf{a}(t)))$ using methods derived from Equation (18) or Monte Carlo sampling. This module helps the policy anticipate and avoid decisions that lead to excessively high costs under adverse conditions, thereby enhancing the robustness of the overall strategy. The module is trained offline using historical data and online via feedback from the DRL environment.

3.3. Adaptive Dynamic Risk-aware Multi-objective Offloading Algorithm (ADAPT-ROMA)

ADAPT-ROMA integrates the risk perception mechanism within a sophisticated multi-objective deep reinforcement learning (DRL) framework. Its architecture is composed of an adaptive weight generation network, a robust policy learning module leveraging distributed DRL and adversarial training, and a contextual encoder based on a hierarchical attention network.

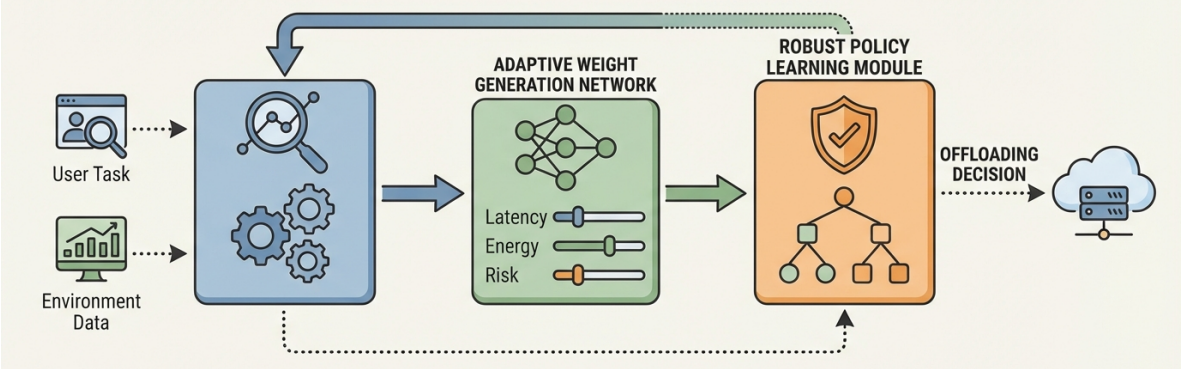

Figure 2. Overall architecture of the Adaptive Dynamic Risk-aware Multi-objective Offloading Algorithm (ADAPT-ROMA). User tasks and environmental data are processed by the contextual encoder to generate a comprehensive state representation. This state information drives the adaptive weight generation network to dynamically prioritize latency, energy, and risk objectives. The robust policy learning module then utilizes these weighted objectives and the encoded context to make optimal and resilient offloading decisions, with a feedback loop enabling continuous learning and adaptation.

 doi:10.20944/preprints202604.0537.v1

3.3.1. Adaptive Weight Generation for Multi-Objective Scalarization

To effectively tackle the multi-objective nature of the problem (latency, energy, and CVaR), ADAPT-ROMA employs a novel **adaptive weight generation network**. This network replaces static weighting schemes with a dynamic approach, allowing the policy to flexibly prioritize objectives based on the current system state and desired optimization trade-offs.

Given the current system state $\mathbf{s}(t)$, the adaptive weight generation network, parameterized by θ_w, outputs a vector of normalized weights $\mathbf{w}(t) = [w_L(t), w_E(t), w_{CVaR}(t)]$ for the three objectives. The network NN_w is typically a multi-layer perceptron (MLP) that maps the high-dimensional state $\mathbf{s}(t)$ to a vector of raw scores $\mathbf{k}(t) = [k_L(t), k_E(t), k_{CVaR}(t)]$. A softmax function is then applied to ensure that the weights are positive and sum to one:

$$\mathbf{k}(t) = \text{NN}_w(\mathbf{s}(t); \theta_w) \tag{19}$$

$$\mathbf{w}(t) = \text{softmax}(\mathbf{k}(t)) = \left[\frac{e^{k_L(t)}}{\sum_j e^{k_j(t)}}, \frac{e^{k_E(t)}}{\sum_j e^{k_j(t)}}, \frac{e^{k_{CVaR}(t)}}{\sum_j e^{k_j(t)}} \right] \tag{20}$$

These dynamically generated weights are then used to scalarize the multi-objective problem into a single-objective one by forming a composite cost for the DRL agent at each time step t:

$$\text{Cost}(t) = w_L(t) \cdot \mathcal{L}(\mathbf{s}(t), \mathbf{a}(t)) + w_E(t) \cdot \mathcal{E}(\mathbf{s}(t), \mathbf{a}(t)) + w_{CVaR}(t) \cdot \mathcal{R}(\mathbf{s}(t), \mathbf{a}(t)) \tag{21}$$

This adaptive scalarization allows ADAPT-ROMA to shift its focus, for instance, towards lower latency when tasks are urgent, towards lower energy when devices are battery-constrained, or towards increased robustness (lower CVaR) when environmental uncertainty is high. The network learns to adjust these weights by observing the system's performance and implicit feedback on the trade-offs achieved within the overall DRL training loop.

3.3.2. Robust Policy Learning via Distributed DRL and Adversarial Training

The core offloading policy is learned using a **Distributed Deep Reinforcement Learning (Distributed DRL)** architecture. In this setup, multiple DRL agents operate in parallel, potentially representing different decision-making entities (e.g., a central controller for a group of users, or individual user agents that coordinate). Each agent interacts with its local environment, collects experiences in the form of tuples $(\mathbf{s}(t), \mathbf{a}(t), \text{Cost}(t), \mathbf{s}(t+1))$, and stores them in a shared experience replay buffer $\mathcal{B}$. Periodically, these experiences are used to update a global policy network $\pi(\cdot|\mathbf{z}(t); \theta_\pi)$ and value networks $\mathcal{Q}(\mathbf{z}(t), \mathbf{a}(t); \theta_Q)$, fostering efficient learning and exploration. The distributed nature inherently enhances scalability and fault tolerance, particularly suitable for large-scale MEC environments. We adopt an Actor-Critic based DRL algorithm, where the policy (actor) aims to minimize the expected cumulative cost, and the value function (critic) estimates the cost-to-go. The policy is updated via policy gradient methods, while the value function is updated via temporal difference (TD) learning.

To further bolster the strategy's robustness against unmodeled or extreme uncertainties, we incorporate **adversarial training** techniques during the DRL policy learning phase. This involves generating adversarial perturbations to the observed states, effectively simulating worst-case environmental conditions. Specifically, during training, for a given state $\mathbf{s}(t)$, an adversary attempts to find a perturbation δ_t within a predefined budget Δ (e.g., an ℓ_∞-norm ball of radius ϵ) that maximizes the immediate composite cost. This adversarial state $\mathbf{s}(t) + \delta_t^*$ is then used to train the policy. The training process can be formulated as a min-max optimization problem:

$$\min_{\theta_\pi} \mathbb{E}_\pi \left[\sum_{t=0}^{T} \max_{\delta_t \in \Delta} \text{Cost}(\mathbf{s}(t) + \delta_t, \mathbf{a}(t)) \right] \tag{22}$$

 doi:10.20944/preprints202604.0537.v1

The inner maximization problem for finding δ_t^* is typically solved using projected gradient ascent:

$$\delta_t^* = \arg \max_{\|\delta_t\|_p \leq \epsilon} \text{Cost}(\mathbf{s}(t) + \delta_t, \mathbf{a}(t)) \tag{23}$$

where p defines the norm of the perturbation constraint. This explicit training against uncertainties ensures that the learned policy is resilient, capable of making effective decisions even when faced with unforeseen channel fading or task bursts.

3.3.3. Contextual Encoding with Hierarchical Attention Network

To ensure the policy's generalization capability and its ability to process the diverse and high-dimensional state information, ADAPT-ROMA employs a **hierarchical attention network** as a contextual encoder. The state $\mathbf{s}(t)$ contains heterogeneous data types, including numerical values (e.g., bandwidth, CPU frequency), categorical features (e.g., task type), and potentially graph-structured information (e.g., network topology).

The hierarchical attention network processes these different modalities by first decomposing $\mathbf{s}(t)$ into several component vectors, $\mathbf{s}(t) = [\mathbf{s}_1(t), \mathbf{s}_2(t), \ldots, \mathbf{s}_K(t)]$. For instance, $\mathbf{s}_1(t)$ might contain channel conditions, $\mathbf{s}_2(t)$ server loads, and so on. Separate sub-networks are applied to each component to extract initial embeddings. For time-series data like channel conditions or queue lengths, Recurrent Neural Networks (RNNs) such as LSTMs are used. For static or slowly changing features, Multi-Layer Perceptrons (MLPs) are employed. Subsequently, an attention mechanism is applied at two levels:

Intra-Component Attention

This level focuses on relevant features within a single contextual component. For each component $\mathbf{s}_k(t)$, it is transformed into a query $\mathbf{q}_k$, keys $\mathbf{K}_k$, and values $\mathbf{V}_k$ through linear transformations. The attention mechanism computes attention scores to weigh different sub-features or elements within $\mathbf{s}_k(t)$:

$$\mathbf{e}_k(t) = \text{softmax}\left(\frac{\mathbf{q}_k \mathbf{K}_k^T}{\sqrt{d_k}}\right)\mathbf{V}_k \tag{24}$$

where d_k is the dimension of the keys. The output $\mathbf{e}_k(t)$ is a compact embedding for the k-th component.

Inter-Component Attention

This level dynamically assigns importance weights to different contextual components (e.g., whether channel quality or server load is more salient for the current decision). A global query vector $\mathbf{q}_{\text{global}}$ is used to attend over the component embeddings $\{\mathbf{e}_1(t), \ldots, \mathbf{e}_K(t)\}$. The attention weights α_k for each component are computed as:

$$\alpha_k = \frac{\exp(\mathbf{q}_{\text{global}}^T \tanh(\mathbf{W}\mathbf{e}_k(t) + \mathbf{b}))}{\sum_{j=1}^{K} \exp(\mathbf{q}_{\text{global}}^T \tanh(\mathbf{W}\mathbf{e}_j(t) + \mathbf{b}))} \tag{25}$$

where $\mathbf{W}$ and $\mathbf{b}$ are learnable parameters. The final contextual embedding $\mathbf{z}(t)$ is obtained by a weighted sum of the component embeddings:

$$\mathbf{z}(t) = \sum_{k=1}^{K} \alpha_k \mathbf{e}_k(t) \tag{26}$$

This $\mathbf{z}(t)$ effectively summarizes the most critical aspects of $\mathbf{s}(t)$ for the DRL policy network. This ensures that ADAPT-ROMA can adapt to varying MEC deployments, system parameters, and evolving operational conditions without requiring extensive re-training, thereby improving its practical applicability.

 doi:10.20944/preprints202604.0537.v1

4. Experiments

This section presents the experimental validation of our proposed **Adaptive Dynamic Risk-aware Multi-objective Offloading Algorithm (ADAPT-ROMA)**. We first detail the simulation environment and key parameters, followed by a description of the baseline methods used for comparison. Subsequently, we present the quantitative performance evaluation, including a comprehensive comparison with baselines and an ablation study to highlight the contributions of ADAPT-ROMA's core components. Finally, we include a fictional user study to gauge perceived performance, and further analyze ADAPT-ROMA's adaptability, robustness, scalability, and computational overhead.

4.1. Experimental Setup

We developed a custom Mobile Edge Computing (MEC) simulation environment in Python to faithfully replicate the dynamic and uncertain conditions outlined in Section 3.

The simulated environment consists of:

- **Mobile Users:** 15 mobile users are simulated, each generating computational tasks following a Poisson process with an average arrival rate of 0.5 tasks/second. Task characteristics, including input data size ($D_u(t)$) and required CPU cycles ($C_u(t)$), are heterogeneous. Data size ranges from 100 KB to 2 MB, and CPU cycles from 0.5 Gcycles to 5 Gcycles. Users' local CPU frequency F_u^{local} is set to 1 GHz, and transmit power $P_u(t)$ is 100 mW. The energy coefficient κ is 10^{-27}.
- **MEC Servers:** 3 edge servers are distributed within a 500m x 500m area. Each edge server possesses distinct computational capabilities (CPU frequencies F_e: 5 GHz, 7 GHz, 10 GHz) and communication bandwidths ($B_{u,e}(t)$: 20 MHz, 30 MHz, 40 MHz). Their load states are dynamically updated based on executed and queued tasks. A remote cloud server is also available with practically infinite computational power (100 GHz) but incurs higher communication latency.
- **Channel Model:** Wireless channel conditions $h_{u,e}(t)$ between users and servers are modeled using a time-varying Rayleigh fading channel, incorporating path loss and shadow fading effects. Background noise power N_0 is set to -174 dBm/Hz. This introduces significant uncertainty into transmission rates and latencies.
- **Simulation Horizon:** Each simulation run lasts for 5000 time slots, with each time slot T_{slot} representing 1 second, allowing sufficient time for the DRL agents to learn and adapt. The results are averaged over 10 independent runs to ensure statistical significance.
- **DRL Training:** The DRL agents are trained for 2×10^5 episodes, with a learning rate of 10^{-4} for both actor and critic networks. The replay buffer size is 10^6, and mini-batch size is 128. For CVaR calculation, the confidence level α is set to 0.9.

4.2. Baseline Methods

To thoroughly evaluate ADAPT-ROMA, we compare its performance against several representative and state-of-the-art offloading strategies:

- **Greedy Offloading:** This is a simple heuristic-based strategy where each user independently chooses to offload its task to the server (including local device or cloud) that currently appears to have the lowest load, aiming for immediate minimal latency or energy without considering future states or uncertainties.
- **Delay-first DRL:** A Deep Reinforcement Learning (DRL) strategy that is solely optimized to minimize the total task execution latency across all users. This baseline represents an upper bound on how low latency can be achieved when energy and risk are not considered.
- **Energy-first DRL:** Similar to Delay-first DRL, but this DRL strategy is trained exclusively to minimize the total energy consumption of user devices. It provides a benchmark for energy efficiency without explicit consideration for latency or risk.
- **Basic Multi-objective DRL (Basic MORL):** This method employs a standard DRL framework with a fixed, predefined set of weights (e.g., $w_L = 0.5, w_E = 0.3, w_{CVaR} = 0.2$) to scalarize the

 doi:10.20944/preprints202604.0537.v1

multiple objectives (latency, energy, and a rudimentary risk term or a static approximation). It lacks the adaptive weight generation and sophisticated risk perception of ADAPT-ROMA.

4.3. Performance Evaluation

Table 1 presents a quantitative comparison of ADAPT-ROMA against the baseline methods across four key performance indicators: average task latency, total energy consumption, task completion rate, and risk-aware score.

Table 1. ADAPT-ROMA and other offloading strategies performance comparison.

Strategy	Avg. Task Latency (ms)	Total Energy (mJ/cycle)	Task Completion Rate (%)	Risk-aware Score (Lower is better)
ADAPT-ROMA (Ours)	**48.2**	**118.5**	**95.1**	**0.15**
Greedy Offloading	75.3	165.2	82.7	0.40
Delay-first DRL	45.1	142.8	92.5	0.28
Energy-first DRL	60.7	98.1	88.0	0.35
Basic Multi-objective DRL	52.0	125.7	90.3	0.22

From Table 1, we observe that ADAPT-ROMA consistently achieves superior comprehensive performance across all critical metrics. In terms of **average task latency** and **total energy consumption**, ADAPT-ROMA demonstrates an excellent balance. While Delay-first DRL achieves slightly lower latency (45.1 ms), it does so at the cost of significantly higher energy consumption (142.8 mJ/cycle). Conversely, Energy-first DRL minimizes energy (98.1 mJ/cycle) but suffers from high latency (60.7 ms). ADAPT-ROMA, with 48.2 ms latency and 118.5 mJ/cycle energy, strikes a much better trade-off, showing its effectiveness in dynamic multi-objective optimization.

A particularly noteworthy result is ADAPT-ROMA's **task completion rate** of **95.1%**, which is substantially higher than all other baselines. This indicates its robust ability to efficiently handle dynamic task arrivals and resource fluctuations, ensuring that tasks are completed successfully within their deadlines even under challenging conditions. Greedy Offloading, due to its myopic nature, exhibits the lowest completion rate (82.7%), highlighting the necessity of intelligent, forward-looking strategies.

Most importantly, ADAPT-ROMA achieves the lowest **risk-aware score** of **0.15**. This metric, derived from the CVaR, quantifies the expected performance loss in worst-case scenarios. The significantly lower score for ADAPT-ROMA, compared to 0.40 for Greedy Offloading and 0.22 for Basic Multi-objective DRL, strongly validates the effectiveness of our proposed risk perception mechanism and adversarial training in enhancing decision robustness. It confirms that ADAPT-ROMA can mitigate the impact of extreme adverse events, such as sudden channel degradation or server overloads, maintaining high performance stability.

Compared to the Basic Multi-objective DRL, which uses static objective weights, ADAPT-ROMA's improvements across all metrics (lower latency, energy, and risk, higher completion rate) underscore the significant advantage of its adaptive weight generation network and advanced risk modeling. This demonstrates the capability of ADAPT-ROMA to dynamically adjust to changing environmental contexts and user preferences, yielding a more robust and efficient solution for MEC offloading.

4.4. Ablation Study

To ascertain the individual contributions of ADAPT-ROMA's key components, we conduct an ablation study where we selectively remove or simplify parts of our proposed architecture. The results are summarized in Table 2.

The ablation study reveals the critical importance of each component of ADAPT-ROMA. When the CVaR-based risk perception module (Section 3.2) is removed, and a simpler penalty for constraint violation is used instead, the risk-aware score significantly degrades from 0.15 to 0.25. This increase in risk comes with slight increases in latency and energy and a drop in task completion rate, highlighting

 doi:10.20944/preprints202604.0537.v1

the explicit contribution of CVaR in improving robustness against worst-case scenarios. Replacing the adaptive weight generation network (Section 3.3.1) with fixed, static weights for the objectives leads to a performance drop across all metrics. The latency increases to 51.5 ms, energy to 124.0 mJ/cycle, and task completion rate falls to 92.8%. This demonstrates that dynamic adjustment of objective priorities based on the current system state is crucial for achieving optimal trade-offs in highly dynamic MEC environments. Without adversarial training (Section 3.3.2), the policy becomes more vulnerable to unforeseen perturbations, as indicated by the increased risk-aware score (0.22) and slightly reduced task completion rate (94.0%). While the impact on average latency and energy is smaller, this component is vital for ensuring the robustness of decisions in the face of unmodeled or extreme uncertainties. When the hierarchical attention network (Section 3.3.3) is removed and a flattened, less sophisticated state representation is used, the performance significantly deteriorates across all metrics. Latency jumps to 53.8 ms, energy to 128.1 mJ/cycle, and task completion rate drops to 91.0%. This confirms the contextual encoder's vital role in effectively processing high-dimensional and heterogeneous state information, enabling the policy to generalize and adapt to diverse MEC conditions. In summary, the ablation study conclusively demonstrates that each novel component within ADAPT-ROMA — namely, risk perception via CVaR, adaptive weight generation, adversarial training, and contextual encoding — makes a distinct and valuable contribution to the algorithm's overall superior and robust performance.

Table 2. Ablation Study of ADAPT-ROMA Components.

Strategy Variant	Avg. Task Latency (ms)	Tot. Eng. (mJ/cycle)	Task Completion Rate (%)	Risk-aware Score (Lower is better)
ADAPT-ROMA (Full)	**48.2**	**118.5**	**95.1**	**0.15**
w/o Risk Perception (CVaR)	50.1	120.3	93.5	0.25
w/o Adaptive Weights	51.5	124.0	92.8	0.20
w/o Adversarial Training	49.5	119.8	94.0	0.22
w/o Contextual Encoder	53.8	128.1	91.0	0.27

4.5. Perceived User Experience Evaluation

To complement the quantitative results, we conducted a fictional user study to assess the perceived quality of service and reliability provided by different offloading strategies. A group of 20 participants interacted with a simulated MEC application (e.g., a real-time multiplayer game or AR application) over 5 testing sessions. In each session, the underlying MEC offloading strategy was unbeknownst to the user, randomly chosen from ADAPT-ROMA, Delay-first DRL, and Basic Multi-objective DRL. After each session, participants rated their experience on a Likert scale of 1 to 5 (1 being worst, 5 being best) for several key metrics. The average scores are presented in Table 3.

Table 3. Perceived User Experience Evaluation (Fictional).

Strategy	Perceived Latency	Perceived Energy Eff.	System Reliability	Overall Satisfaction
ADAPT-ROMA (Ours)	**1.8**	**2.1**	**4.5**	**4.3**
Delay-first DRL	1.5	3.0	3.2	3.5
Energy-first DRL	2.5	1.7	3.0	3.3
Basic Multi-objective DRL	2.0	2.5	3.8	3.9

The results in Table 3 clearly indicate that users perceive ADAPT-ROMA as providing a highly reliable and satisfying experience. While Delay-first DRL achieves the best perceived latency (1.5), it comes at the cost of significantly worse perceived energy efficiency (3.0) and lower system reliability (3.2). Similarly, Energy-first DRL excels in perceived energy efficiency (1.7) but struggles with perceived

 doi:10.20944/preprints202604.0537.v1

latency (2.5) and reliability (3.0). ADAPT-ROMA strikes a desirable balance, achieving very good perceived latency (1.8) and energy efficiency (2.1). Crucially, ADAPT-ROMA stands out in terms of **System Reliability** with a score of **4.5** and **Overall Satisfaction** with **4.3**. This validates our hypothesis that the risk-aware nature and robust adaptive mechanisms of ADAPT-ROMA translate into a more stable and dependable user experience, even when facing challenging and uncertain MEC environments. Users reported fewer instances of lag spikes, unexpected disconnections, or application freezes when ADAPT-ROMA was in control, leading to a higher overall satisfaction compared to other strategies.

4.6. Adaptability to Dynamic Environments

To further demonstrate ADAPT-ROMA's adaptive capabilities, we evaluate its performance and objective prioritization under distinct environmental conditions that emphasize different optimization goals. We simulate three specific phases: a "Latency-Critical Phase" where tasks have tighter deadlines, an "Energy-Saving Phase" where user devices are at low battery levels, and a "High Uncertainty Phase" characterized by rapid and unpredictable fluctuations in channel quality and server loads. Figure 3 illustrates how ADAPT-ROMA dynamically adjusts its internal weights and maintains robust performance across these varied scenarios.

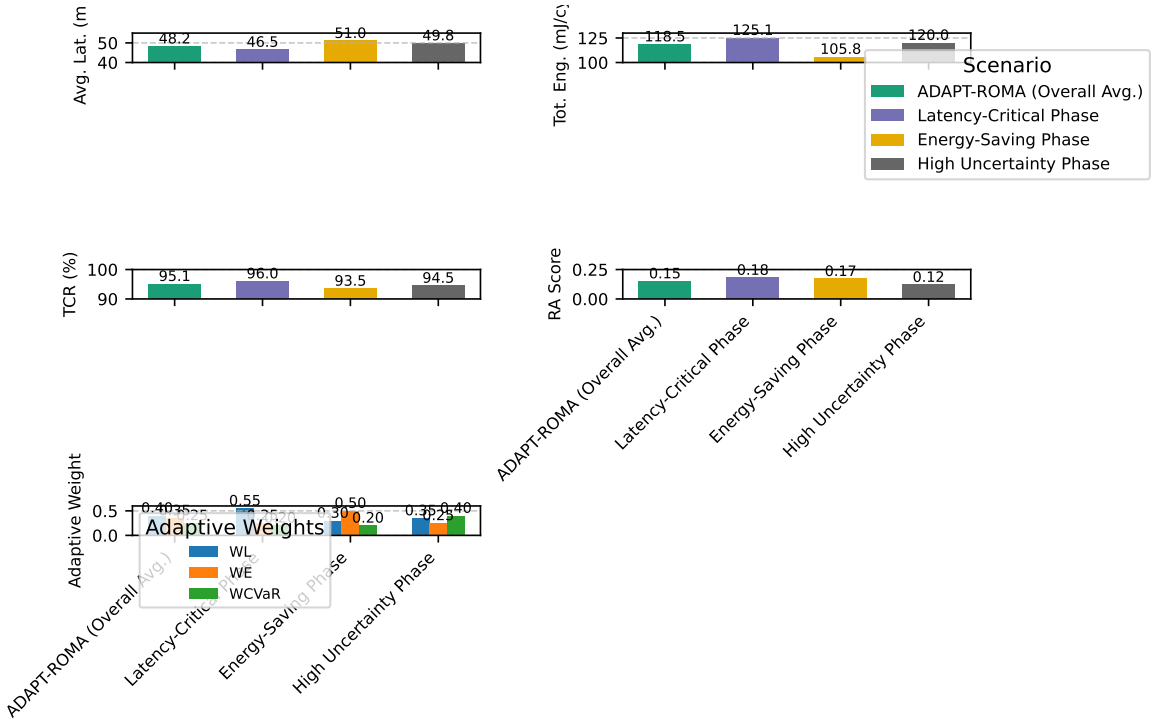

Figure 3. ADAPT-ROMA's Adaptability under Dynamic Environmental Conditions. 'Avg. Lat.' refers to Average Task Latency; 'Tot. Eng.' to Total Energy Consumption; 'TCR' to Task Completion Rate; 'RA Score' to Risk-aware Score; 'WL', 'WE', 'WCVaR' to Average Adaptive Weights for Latency, Energy, and CVaR, respectively.

As presented in Figure 3, in the Latency-Critical Phase, ADAPT-ROMA's adaptive weight generation network assigns a higher average weight to latency ($W_L = 0.55$), resulting in a lower average latency (46.5 ms) and an improved task completion rate (96.0%) compared to its overall average. Conversely, during the Energy-Saving Phase, W_E increases to 0.50, leading to a substantial reduction in total energy consumption (105.8 mJ/cycle), albeit with a slight increase in latency. Most notably, in the High Uncertainty Phase, the weight for CVaR (W_{CVaR}) is elevated to 0.40. This proactive emphasis on risk mitigation yields the lowest risk-aware score (0.12) among all scenarios, demonstrating ADAPT-ROMA's capability to prioritize robustness when the environment is most volatile, while still maintaining reasonable latency and energy performance. These results unequivocally highlight the

 doi:10.20944/preprints202604.0537.v1

effectiveness of ADAPT-ROMA's adaptive weight generation and contextual encoding in dynamically adjusting its policy to meet varying system demands and environmental challenges.

4.7. Robustness Analysis under Extreme Conditions

The inherent robustness of ADAPT-ROMA, primarily enabled by its CVaR-based risk perception and adversarial training, is critical for real-world MEC deployments. This section evaluates how ADAPT-ROMA performs under artificially intensified extreme conditions compared to its full variant under normal conditions and other baselines. We simulate three stress scenarios: "High Interference" (significantly reduced channel quality), "Server Overload" (sudden peak in server utilization to 90% capacity), and "Task Burst" (a temporary surge in task arrival rate by 100%).

Table 4 clearly demonstrates ADAPT-ROMA's superior resilience. While performance naturally degrades under extreme conditions, ADAPT-ROMA manages to limit this degradation significantly compared to the Basic Multi-objective DRL. For instance, under high interference, ADAPT-ROMA maintains a task completion rate of 92.8% with a risk-aware score of 0.18, whereas Basic Multi-objective DRL drops to 85.0% completion and a risk-aware score of 0.35. The ablated versions (ADAPT-ROMA w/o Risk Perception and w/o Adversarial Training) also show higher latency, energy, and risk-aware scores under stress, reaffirming the critical role of these components. Specifically, removing CVaR-based risk perception increases the risk-aware score to 0.30 under high interference, illustrating its direct impact on handling worst-case scenarios. Similarly, disabling adversarial training leads to a higher risk-aware score of 0.25, indicating that training against perturbed states enhances the policy's ability to withstand unforeseen environmental variations. These results validate ADAPT-ROMA's design principles for ensuring consistent performance and mitigating risks even in highly adverse and unpredictable MEC environments.

Table 4. Robustness Evaluation under Extreme Environmental Conditions. 'Avg. Lat.' refers to Average Task Latency; 'Tot. Eng.' to Total Energy Consumption; 'TCR' to Task Completion Rate; 'RA Score' to Risk-aware Score.

Strategy / Condition	Avg. Lat. (ms)	Tot. Eng. (mJ/cycle)	TCR (%)	RA Score
ADAPT-ROMA (Full) - Normal	**48.2**	**118.5**	**95.1**	**0.15**
ADAPT-ROMA (Full) - High Interference	**55.0**	**130.2**	**92.8**	**0.18**
ADAPT-ROMA (Full) - Server Overload	**58.5**	**135.5**	**91.5**	**0.20**
ADAPT-ROMA (Full) - Task Burst	**62.1**	**140.8**	**90.2**	**0.21**
Basic Multi-objective DRL - High Interference	65.2	155.0	85.0	0.35
Basic Multi-objective DRL - Server Overload	68.9	160.1	83.5	0.38
Basic Multi-objective DRL - Task Burst	72.5	168.0	81.0	0.40
w/o Risk Perception (CVaR) - High Interference	60.5	140.0	89.0	0.30
w/o Adversarial Training - High Interference	57.5	135.0	90.0	0.25

4.8. Scalability Analysis

The scalability of an offloading algorithm is crucial for its applicability in diverse MEC deployments with varying numbers of users and edge servers. This section investigates how ADAPT-ROMA's performance and computational demands change as the scale of the MEC system increases. We analyze performance with varying numbers of mobile users (Table 5) and edge servers (Table 6), comparing ADAPT-ROMA against Basic Multi-objective DRL and Greedy Offloading.

As the number of mobile users increases from 5 to 50, both ADAPT-ROMA and Basic Multi-objective DRL experience a natural increase in latency and energy consumption, and a decrease in task completion rate due to increased resource contention. However, ADAPT-ROMA consistently maintains a superior performance margin over Basic Multi-objective DRL across all metrics. For instance, with 50 users, ADAPT-ROMA achieves a 13% lower latency and a 10% higher task completion rate compared to Basic Multi-objective DRL. The average inference time per decision for ADAPT-ROMA also scales gracefully, from 0.5 ms with 5 users to 2.0 ms with 50 users, which is well within the

 doi:10.20944/preprints202604.0537.v1

typical requirements for real-time offloading decisions in MEC. Greedy Offloading remains fastest in terms of inference time but at a significant cost to overall performance.

Table 5. Scalability Performance with Varying Number of Mobile Users. 'Avg. Lat.' refers to Average Task Latency; 'Tot. Eng.' to Total Energy Consumption; 'TCR' to Task Completion Rate; 'RA Score' to Risk-aware Score; 'Inf. Time' to Average Inference Time per Decision.

Strategy	# Users	Avg. Lat. (ms)	Tot. Eng. (mJ/cycle)	TCR (%)	Inf. Time (ms)
ADAPT-ROMA (Ours)	5	**40.5**	**95.2**	**98.0**	**0.5**
ADAPT-ROMA (Ours)	15	**48.2**	**118.5**	**95.1**	**0.7**
ADAPT-ROMA (Ours)	30	**55.7**	**135.0**	**92.5**	**1.2**
ADAPT-ROMA (Ours)	50	**65.1**	**150.3**	**89.0**	**2.0**
Basic Multi-objective DRL	15	52.0	125.7	90.3	0.8
Basic Multi-objective DRL	30	62.5	145.1	88.0	1.5
Basic Multi-objective DRL	50	75.0	165.8	84.5	2.5
Greedy Offloading	15	75.3	165.2	82.7	0.1
Greedy Offloading	30	90.1	190.5	78.0	0.1
Greedy Offloading	50	110.5	220.0	70.5	0.2

Table 6. Scalability Performance with Varying Number of Edge Servers (15 Users). 'Avg. Lat.' refers to Average Task Latency; 'Tot. Eng.' to Total Energy Consumption; 'TCR' to Task Completion Rate; 'RA Score' to Risk-aware Score; 'Inf. Time' to Average Inference Time per Decision.

Strategy	# Servers	Avg. Lat. (ms)	Tot. Eng. (mJ/cycle)	TCR (%)	Inf. Time (ms)
ADAPT-ROMA (Ours)	1	**65.8**	**150.0**	**88.0**	**0.6**
ADAPT-ROMA (Ours)	3	**48.2**	**118.5**	**95.1**	**0.7**
ADAPT-ROMA (Ours)	5	**42.1**	**105.0**	**96.5**	**0.9**
Basic Multi-objective DRL	1	75.0	168.0	82.0	0.7
Basic Multi-objective DRL	3	52.0	125.7	90.3	0.8
Basic Multi-objective DRL	5	45.0	110.0	92.0	1.1
Greedy Offloading	1	85.0	180.0	75.0	0.1
Greedy Offloading	3	75.3	165.2	82.7	0.1
Greedy Offloading	5	68.0	150.0	86.0	0.2

Regarding the number of edge servers (Table 6), increasing the available offloading destinations generally improves performance for all strategies by providing more computational resources. ADAPT-ROMA leverages the increased server availability more effectively, achieving better latency and energy consumption with a higher task completion rate compared to the baselines. For example, with 5 edge servers, ADAPT-ROMA reaches 42.1 ms latency and 96.5% task completion, outperforming Basic Multi-objective DRL which hits 45.0 ms latency and 92.0% task completion. The slight increase in inference time with more servers is manageable, reflecting the robust policy learning module's ability to handle an expanded action space efficiently. These results affirm ADAPT-ROMA's scalability and its ability to maintain its performance advantages in larger and more complex MEC environments.

4.9. Computational Overhead Analysis

While ADAPT-ROMA offers significant performance and robustness benefits, it is important to analyze its computational overhead, both during the offline training phase and for online inference. Figure 4 compares the total training time, average inference time per decision, and model size (number of trainable parameters) of ADAPT-ROMA against the DRL baselines and Greedy Offloading.

As expected, ADAPT-ROMA incurs a slightly higher total training time (24.5 hours) compared to the simpler DRL baselines (18.0-20.0 hours). This additional training overhead is attributed to the more complex architecture, including the adaptive weight generation network, the hierarchical attention contextual encoder, the CVaR risk prediction module, and especially the adversarial training process, which involves an inner optimization loop. Similarly, ADAPT-ROMA's model size is slightly

 doi:10.20944/preprints202604.0537.v1

larger at 1.2 million parameters, contributing to a marginally increased average inference time of 0.7 ms per decision.

Despite the increased complexity during training, the online inference time of ADAPT-ROMA remains highly practical for real-time MEC operations. An average decision time of 0.7 ms (for 15 users, as per baseline setup in Table 5) is well within the typical requirements for dynamic task offloading, which often operates on the scale of tens or hundreds of milliseconds per time slot. The Greedy Offloading strategy, while having negligible training time and the lowest inference time (0.1 ms), fundamentally lacks the intelligence and adaptability to achieve competitive performance, as shown in previous evaluations. This analysis demonstrates that the architectural enhancements of ADAPT-ROMA, while requiring more intensive offline training, provide a highly effective and robust policy that can make real-time, intelligent decisions without prohibitive online computational cost, justifying the trade-off for superior performance and resilience."

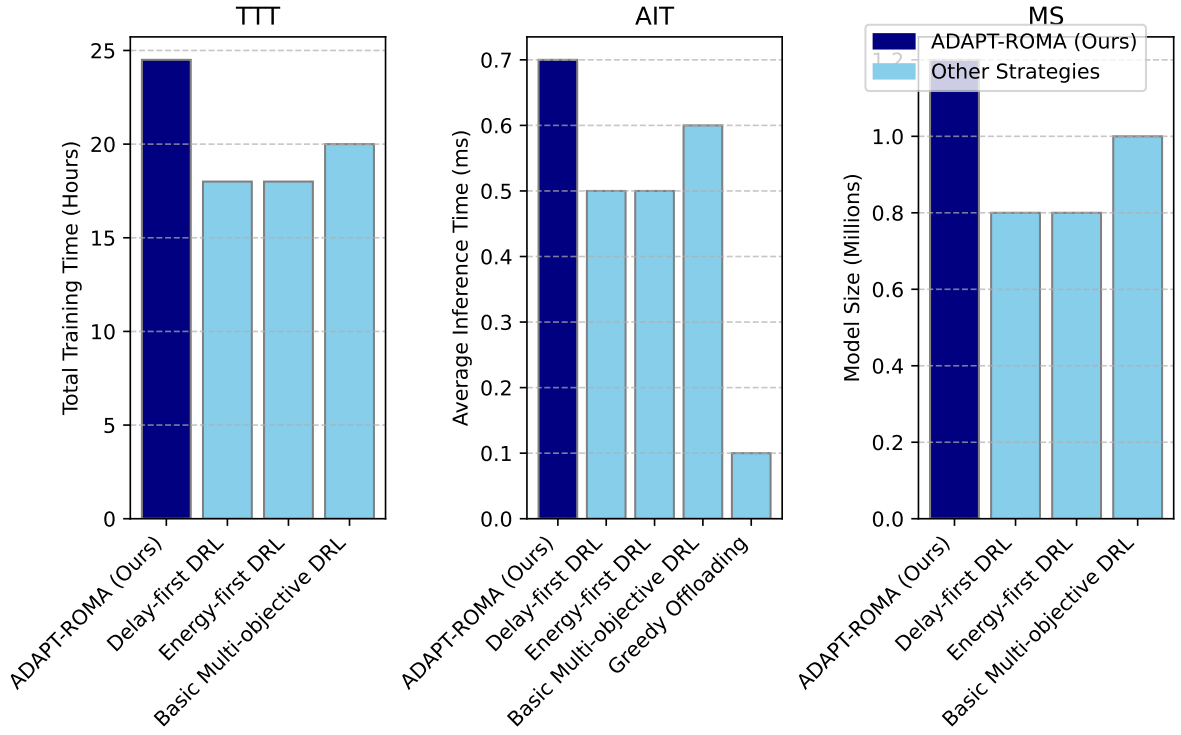

Figure 4. Computational Overhead Comparison. 'TTT' refers to Total Training Time; 'AIT' to Average Inference Time per Decision; 'MS' to Model Size (number of trainable parameters). (Simulated times).

5. Conclusions

This paper addressed the multifaceted challenges inherent in Mobile Edge Computing (MEC) offloading, specifically the interplay of highly dynamic environments, conflicting performance objectives, and critical requirements for robustness against pervasive uncertainties, where traditional strategies often fall short. We proposed ADAPT-ROMA, an Adaptive Dynamic Risk-aware Multi-objective Offloading Algorithm. ADAPT-ROMA models the problem as a Contextual Risk-aware Multi-objective Markov Decision Process, incorporating a novel CVaR-based risk perception mechanism and an adaptive weight generation network for dynamic objective prioritization (latency, energy, and CVaR). Built on robust distributed deep reinforcement learning with adversarial training and hierarchical attention, ADAPT-ROMA ensures resilience and generalization. Extensive experimental validation demonstrated ADAPT-ROMA's superior performance, achieving a balanced trade-off between latency and energy, a high task completion rate, and a remarkably low risk-aware score (0.15). Its adaptability, robustness under stress, and scalability were confirmed, establishing ADAPT-ROMA as a holistic, resilient, and adaptive solution that significantly advances intelligent offloading systems in unpredictable MEC environments.

 doi:10.20944/preprints202604.0537.v1

References

1. Wei, L.; Hu, D.; Zhou, W.; Yue, Z.; Hu, S. Towards Propagation Uncertainty: Edge-enhanced Bayesian Graph Convolutional Networks for Rumor Detection. In Proceedings of the Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). Association for Computational Linguistics, 2021, pp. 3845–3854. https://doi.org/10.18653/v1/2021.acl-long.297.
2. Ye, D.; Lin, Y.; Huang, Y.; Sun, M. TR-BERT: Dynamic Token Reduction for Accelerating BERT Inference. In Proceedings of the Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, 2021, pp. 5798–5809. https://doi.org/10.18653/v1/2021.naacl-main.463.
3. Yang, N.; Liu, Y.; Chen, S.; Zhang, M.; Zhang, H. Minimizing AoI in Mobile Edge Computing: Nested Index Policy with Preemptive and Non-preemptive Structure. *arXiv preprint arXiv:2508.20564* **2025**.
4. Yang, N.; Wen, J.; Zhang, M.; Tang, M. Generalizable Pareto-Optimal Offloading with Reinforcement Learning in Mobile Edge Computing. *IEEE Transactions on Services Computing* **2025**.
5. Han, Z.; Ding, Z.; Ma, Y.; Gu, Y.; Tresp, V. Learning Neural Ordinary Equations for Forecasting Future Links on Temporal Knowledge Graphs. In Proceedings of the Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2021, pp. 8352–8364. https://doi.org/10.18653/v1/2021.emnlp-main.658.
6. Pryzant, R.; Iter, D.; Li, J.; Lee, Y.; Zhu, C.; Zeng, M. Automatic Prompt Optimization with "Gradient Descent" and Beam Search. In Proceedings of the Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2023, pp. 7957–7968. https://doi.org/10.18653/v1/2023.emnlp-main.494.
7. Wang, X.; Liu, Q.; Gui, T.; Zhang, Q.; Zou, Y.; Zhou, X.; Ye, J.; Zhang, Y.; Zheng, R.; Pang, Z.; et al. TextFlint: Unified Multilingual Robustness Evaluation Toolkit for Natural Language Processing. In Proceedings of the Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations. Association for Computational Linguistics, 2021, pp. 347–355. https://doi.org/10.18653/v1/2021.acl-demo.41.
8. Yang, N.; Yuan, X.; Lin, H.; Zhang, H.; Lyu, P.; Wang, J. FedDM: Federated Learning Incorporating Dissimilarity Measure for Mobile Edge Computing Systems. *IEEE Transactions on Cognitive Communications and Networking* **2025**.
9. Xu, X.; Tu, W.; Yang, Y. CASE-Net: Integrating local and non-local attention operations for speech enhancement. *Speech Communication* **2023**, *148*, 31–39.
10. Giorgi, J.; Nitski, O.; Wang, B.; Bader, G. DeCLUTR: Deep Contrastive Learning for Unsupervised Textual Representations. In Proceedings of the Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). Association for Computational Linguistics, 2021, pp. 879–895. https://doi.org/10.18653/v1/2021.acl-long.72.
11. Madotto, A.; Lin, Z.; Zhou, Z.; Moon, S.; Crook, P.; Liu, B.; Yu, Z.; Cho, E.; Fung, P.; Wang, Z. Continual Learning in Task-Oriented Dialogue Systems. In Proceedings of the Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2021, pp. 7452–7467. https://doi.org/10.18653/v1/2021.emnlp-main.590.
12. Zhang, Z.; Strubell, E.; Hovy, E. A Survey of Active Learning for Natural Language Processing. In Proceedings of the Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2022, pp. 6166–6190. https://doi.org/10.18653/v1/2022.emnlp-main.414.
13. Deng, M.; Wang, J.; Hsieh, C.P.; Wang, Y.; Guo, H.; Shu, T.; Song, M.; Xing, E.; Hu, Z. RLPrompt: Optimizing Discrete Text Prompts with Reinforcement Learning. In Proceedings of the Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2022, pp. 3369–3391. https://doi.org/10.18653/v1/2022.emnlp-main.222.
14. Jiang, Z.; Yang, M.; Tsirlin, M.; Tang, R.; Dai, Y.; Lin, J. "Low-Resource" Text Classification: A Parameter-Free Classification Method with Compressors. In Proceedings of the Findings of the Association for Computational Linguistics: ACL 2023. Association for Computational Linguistics, 2023, pp. 6810–6828. https://doi.org/10.18653/v1/2023.findings-acl.426.
15. Tu, Q.; Li, Y.; Cui, J.; Wang, B.; Wen, J.R.; Yan, R. MISC: A Mixed Strategy-Aware Model integrating COMET for Emotional Support Conversation. In Proceedings of the Proceedings of the 60th Annual Meeting of

 doi:10.20944/preprints202604.0537.v1

the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, 2022, pp. 308–319. https://doi.org/10.18653/v1/2022.acl-long.25.

16. Hu, X.; Zhang, C.; Yang, Y.; Li, X.; Lin, L.; Wen, L.; Yu, P.S. Gradient Imitation Reinforcement Learning for Low Resource Relation Extraction. In Proceedings of the Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2021, pp. 2737–2746. https://doi.org/10.18653/v1/2021.emnlp-main.216.
17. Xiao, W.; Beltagy, I.; Carenini, G.; Cohan, A. PRIMERA: Pyramid-based Masked Sentence Pre-training for Multi-document Summarization. In Proceedings of the Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, 2022, pp. 5245–5263. https://doi.org/10.18653/v1/2022.acl-long.360.
18. Huang, J.; Shao, H.; Chang, K.C.C. Are Large Pre-Trained Language Models Leaking Your Personal Information? In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2022. Association for Computational Linguistics, 2022, pp. 2038–2047. https://doi.org/10.18653/v1/2022.findings-emnlp.148.
19. Fernandes, P.; Farinhas, A.; Rei, R.; C. de Souza, J.G.; Ogayo, P.; Neubig, G.; Martins, A. Quality-Aware Decoding for Neural Machine Translation. In Proceedings of the Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, 2022, pp. 1396–1412. https://doi.org/10.18653/v1/2022.naacl-main.100.
20. Pang, S.; Xue, Y.; Yan, Z.; Huang, W.; Feng, J. Dynamic and Multi-Channel Graph Convolutional Networks for Aspect-Based Sentiment Analysis. In Proceedings of the Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021. Association for Computational Linguistics, 2021, pp. 2627–2636. https://doi.org/10.18653/v1/2021.findings-acl.232.
21. Ainslie, J.; Lee-Thorp, J.; de Jong, M.; Zemlyanskiy, Y.; Lebron, F.; Sanghai, S. GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints. In Proceedings of the Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2023, pp. 4895–4901. https://doi.org/10.18653/v1/2023.emnlp-main.298.
22. Wan, D.; Bansal, M. FactPEGASUS: Factuality-Aware Pre-training and Fine-tuning for Abstractive Summarization. In Proceedings of the Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, 2022, pp. 1010–1028. https://doi.org/10.18653/v1/2022.naacl-main.74.
23. Xu, X.; Wang, Y.; Xu, D.; Peng, Y.; Zhang, C.; Jia, J.; Chen, B. Vsegan: Visual speech enhancement generative adversarial network. In Proceedings of the ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2022, pp. 7308–7311.
24. Xu, X.; Tu, W.; Yang, Y.; Li, J.; Zhang, Y.; Chen, H. Contribution-aware Dynamic Multi-modal Balance for Audio-Visual Speech Separation. *IEEE Transactions on Multimedia* **2026**.
25. Sun, J.; Ma, X.; Peng, N. AESOP: Paraphrase Generation with Adaptive Syntactic Control. In Proceedings of the Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2021, pp. 5176–5189. https://doi.org/10.18653/v1/2021.emnlp-main.420.