



Computing efficiency and latency optimization in intelligent reflecting surface-enhanced device-to-device mobile edge computing via lagrange-dual deep learning

Kimchheang Chhea¹ · Jung-Ryun Lee^{1,2}

Received: 24 September 2025 / Accepted: 13 December 2025 / Published online: 10 January 2026
© The Author(s) 2026

Abstract

In this work, we investigate device-to-device (D2D) mobile edge computing (MEC) with *intelligent reflecting surface (IRS)* aiding both D2D and device-to-MEC links to reduce energy and delay for user devices. We formulate a non-convex computation offloading problem that maximizes computing efficiency over latency under mixed integer, linear, and non-linear constraints and jointly optimize offloading decisions, partial offloading ratio, edge computing frequency, transmit power, and IRS phase shifts via a fully connected deep neural network (DNN) trained with a Lagrange-dual loss function that embeds the objective and inequality constraints. Extensive simulations across varying maximum task sizes (0.5–3 Mbits), edge computing capacities (10–90 Mcycles/s), and number of IRS elements (10–30) show that the proposed method consistently achieves solutions within 5–10% of near-global optimal solutions obtained by exhaustive search (ES) and outperforms gradient search (GS), fixed-offloading DNN and REINFORCE algorithm that achieves approximately 50% higher computing efficiency at $N_1 = 20$ users. The proposed DNN exhibits 3–5% energy consumption gap from ES while maintaining lower computational complexity of $O(12N_c N^2 Ms)$ versus $O(\omega^{-2})$ for GS, $O((2Q_s)^{(N+M)})$ for ES, and $O(\iota^{-4})$ for REINFORCE, which exhibits inference times on the order of milliseconds, compared to the computationally intractable costs associated with ES. These results show that the proposed algorithm for IRS-assisted partial D2D/MEC offloading is an effective and practical approach for jointly enhancing computing efficiency and latency in next-generation wireless systems.

Keywords Mobile edge computing · Device-to-device communication · Intelligent reflecting surface · Partial offloading · Resource allocation · Deep neural network · Computing efficiency

1 Introduction

Next-generation wireless networks are expected to play a crucial role in various emerging applications such as intelligent identification, positioning, and tracking. These applications typically require high processing power and significantly

drain the device batteries (Ferreira et al. 2025). Traditionally, cloud computing, which provides computing services via the Internet from centralized servers, has been proposed to support these computationally intensive tasks, but at the cost of added transmission latency (Chai et al. 2025).

Mobile edge computing (MEC) addresses this challenge by placing computing resources close to the network edge, thereby reducing latency. In MEC networks, users decide how and where tasks are processed by offloading some or all of their tasks to nearby computation access points (Almasaeed 2025), and since this offloading plays a central role in balancing the workload between edge servers and devices, it is a key design factor for reducing latency, lowering energy consumption, and thus ensuring efficient resource use Spinelli and Mancuso (2020).

Although MEC offers significant benefits, its performance can deteriorate when multiple users are connected

✉ Jung-Ryun Lee
jrlee@cau.ac.kr

Kimchheang Chhea
chheangkim@cau.ac.kr

¹ Department of Intelligent Energy and Industry, Chung-Ang University, 84 Heuksuk-ro, Dongjak-gu, Seoul, Seoul 17546, Republic of Korea

² School of Electrical and Electronics Engineering, Chung-Ang University, 84 Heuksuk-ro, Dongjak-gu, Seoul, Seoul 17546, Republic of Korea

simultaneously. Therefore, it is important to develop an optimized offloading strategy that relieves the computing load on the MEC infrastructure under the given network conditions.

While MEC offers enough computing resources for users, high path loss in MEC communications is inevitable due to the distance between devices and the base station (BS) (Cao et al. 2018). Device-to-device (D2D) offloading has been introduced as an effective way to tackle this issue (Wang et al. 2025). It reduces computation time for resource-intensive applications by offloading some tasks to nearby users through D2D links (Liu et al. 2021). This approach allows devices to connect directly without the Internet and leverages unused computing resources in their vicinity, significantly reducing bandwidth consumption.

Although combining MEC and D2D communications can improve system performance by leveraging the energy efficiency of D2D communication and the high computing power of MEC servers, there is an inherent problem of the low transmit power of D2D offloading links. This limitation can be addressed with an intelligent reflecting surface (IRS), a metasurface composed of many passive reflecting elements that can manipulate the phase of electromagnetic waves to reconfigure wireless channels. By creating additional propagation paths for non-line-of-sight (NLoS) links through passive reflection, the IRS can enhance available paths and improve the signal-to-interference-plus-noise ratio (SINR).

On the other hand, the use of artificial intelligence (AI) has developed continuously across various research areas due to its ability to substantially reduce the complexity of computation in complex environments (Sun et al. 2025; Yazici et al. 2023). For example, neural networks (NN) effectively capture complex dependencies (Jin et al. 2025; Jin and Xu 2025a), Gaussian process regression excels in uncertainty quantification (Jin and Xu 2024b, a), graphical techniques address high-dimensional modeling (Jin and Xu 2025b; Xu and Zhang 2022), and ensemble methods enhance robustness (Xu 2020; Xu and Zhang 2021). Among these approaches, NN have gained particular prominence due to their versatility and scalability across diverse applications. Specifically, the integration of NN optimization tools with wireless systems can improve the performance of network functions and effectively reduce decision-making costs (Chen et al. 2024).

Efforts to improve MEC network efficiency have resulted in a body of work that explores different optimization objectives and solution methodologies. Addressing dynamic network conditions, Li et al. (2021) studied partial offloading in time-varying MEC networks by proposing a deep Q-network to jointly tackle latency and energy consumption problems, with simulation results demonstrating performance gains over conventional approaches. Complementing this with application-specific optimization, Seo et al. (2021) investigated energy-efficient mobile augmented reality services

in MEC by explicitly trading off between energy consumption and latency, proposing a theoretical scheme that controls transmit power and cache size to achieve both low energy and latency as validated by their results. While these studies effectively optimize MEC performance through learning-based adaptation and energy-latency balancing respectively, they focus on direct device-to-server offloading without exploring cooperative offloading opportunities through device-level collaboration.

Recognizing BS's limited computational resources and the potential for proximity-based collaboration, researchers have integrated MEC with D2D communications to address computational resource limitations. From a system capacity perspective, He et al. (2019) formulated an optimization problem to maximize the number of supported cellular devices subject to communication and computation resource constraints, with results demonstrating a substantial increase in overall system computation capacity. Shifting focus to cost efficiency, Chai et al. (2019) addressed task execution cost minimization in systems with an MEC server and multiple cellular D2D users, proposing the Kuhn-Munkres algorithm to solve the mode selection and offloading problem, thereby achieving low task execution costs. Advancing toward more comprehensive optimization, Peng et al. (2021) tackled the joint problem of multi-user cooperative partial offloading, transmission scheduling, and computation allocation in D2D underlaid MEC networks, aiming to minimize both network latency and energy consumption with results revealing significantly lower latency compared to benchmark algorithms. While these contributions establish the viability of MEC-D2D integration across multiple performance dimensions including capacity, cost, and latency, they overlook the underlying constraint of diminished communication range and weakened signal reception inherent to the low-power transmission nature of D2D communications.

To address network performance and resource utilization challenges in D2D communications, researchers have progressively integrated IRS technology with varying optimization objectives and system considerations. From a resource allocation perspective, Wang et al. (2021) tackled the fundamental problem of maximizing sum-rate in underlying D2D communications by jointly controlling active and passive beamforming along with transmit power through successive convex methods, demonstrating substantial system throughput improvements. Building upon throughput considerations, Van et al. (2022) extended the analysis to D2D coexistence scenarios where the base station simultaneously transfers information and energy to users, proposing an alternating descent algorithm to solve the max-min throughput problem with results confirming that data rate scales positively with the number of IRS reflecting elements. Complementing these performance optimization studies, Li (2022) addressed the practical design challenge of IRS deployment by minimizing

the number of reflecting elements subject to outage performance and energy consumption constraints in multi-user D2D scenarios, with simulations revealing optimal configurations that vary across different noise and power operating regions. Extending beyond pure communication metrics, Mao et al. (2021) formulated a computing latency minimization problem for D2D cooperative computing systems aided by IRS, achieving significantly lower latency through joint optimization of task assignment, transmit power, bandwidth, and IRS phase-shift compared to conventional approaches. While these studies collectively demonstrate substantial IRS benefits across resource allocation, throughput optimization, system design, and latency reduction, they focus on either D2D communication enhancement or isolated computing scenarios without exploring the synergistic integration of both paradigms within a unified IRS-assisted framework.

Existing works treat MEC, D2D, and IRS separately or in pairs, with no prior study jointly optimizing IRS-assisted D2D-MEC systems with partial offloading. Furthermore, most approaches rely on iterative algorithms with high computational complexity, while learning-based methods remain underexplored for this joint optimization problem. To address these gaps, this paper proposes a DNN with Lagrange-dual loss function that simultaneously optimizes offloading decisions, partial offloading ratios, edge computing frequency, transmit power, and IRS phase shifts. By embedding both the optimization objective and constraints directly into the learning process, the proposed approach efficiently maximizes computing efficiency while minimizing energy consumption and latency.

The ideas and methodology of this study are summarized below.

- We formulate a novel joint optimization problem for IRS-enhanced D2D-MEC systems that maximizes the ratio of computing efficiency to latency, a unified metric that simultaneously captures both energy and delay performance. In contrast to prior works that address MEC, D2D, or IRS in isolation or pairwise combinations, our formulation is the first to jointly optimize offloading decisions (local/D2D/MEC), partial offloading ratios, edge computing frequency allocation, transmit power, and IRS phase shifts under a challenging set of mixed integer, linear, and non-linear constraints.
- We propose a fully connected DNN trained with a novel Lagrange-dual loss function that embeds both the optimization objective and inequality constraints directly into the learning process. This approach enables simultaneous backpropagation-based updates of both primal variables (offloading decisions, partial offloading ratio, edge computing frequency, transmit power, and IRS phase shifts) and dual variables (Lagrange multipliers). The

framework maintains polynomial computational complexity that exhibits scalable growth with network size.

- Extensive simulations across diverse system parameters (task sizes: 0.5-3 Mbits, edge capacity: 10-90 Mcycles/s, IRS elements: 10-30, users: 2-20) demonstrate that the proposed DNN consistently achieves near-optimal computing efficiency, total energy consumption, and total latency. The proposed DNN utilizing a Lagrange-dual loss function demonstrates superior performance in both objective value optimization and convergence rate when benchmarked against fixed penalty and barrier method approaches. The proposed DNN attains solutions within 5-10% of ES benchmarks while outperforming GS by up to 50% in computing efficiency, particularly at higher number of user devices where GS converges to suboptimal local minima. Additionally, the proposed DNN that uses a partial offloading strategy reduces energy consumption by 15-25% compared to fixed full-offloading approaches and achieves lower latency than the REINFORCE algorithm, which suffers from high gradient variance. Critically, the DNN maintains scalability with polynomial complexity, enabling real-time decision-making in scenarios where ES becomes computationally intractable.

The remainder of this paper is organized as follows. In Section 2, we present the proposed system, channel models, and problem formulation. Section 3 describes iterative methods such as exhaustive search (ES) and GS. The proposed DNN algorithm is presented in Section 4. Section 5 illustrates the numerical results to evaluate the performance of the proposed scheme. Finally, we conclude the paper by providing a summary and discussion of limitations and future work in Section 6.

2 System model and problem formulation

In this study, we consider a wireless system (Fig. 1) consisting of a BS connected to a MEC server via high-speed optical fiber (Tian et al. 2024) and N randomly deployed D2D pairs, denoted by the set $\mathcal{N} = \{1, 2, \dots, N\}$, within the BS's coverage area. An IRS with M reflecting elements, arranged in M_r rows and M_c columns, is deployed to support both D2D and device-to-MEC communications. The devices are divided into two groups. Group 1, denoted as $\mathcal{N}_1 = \{1, \dots, N_1\}$, includes N_1 user devices that need to offload their tasks due to limited computing capability (Yang et al. 2025). Group 2, represented as $\mathcal{N}_2 = \{1, \dots, N_2\}$, consists of N_2 helper devices with spare computing resources. For simplicity, it is assumed that each user device i in $\mathcal{N}_1$ is paired with a device

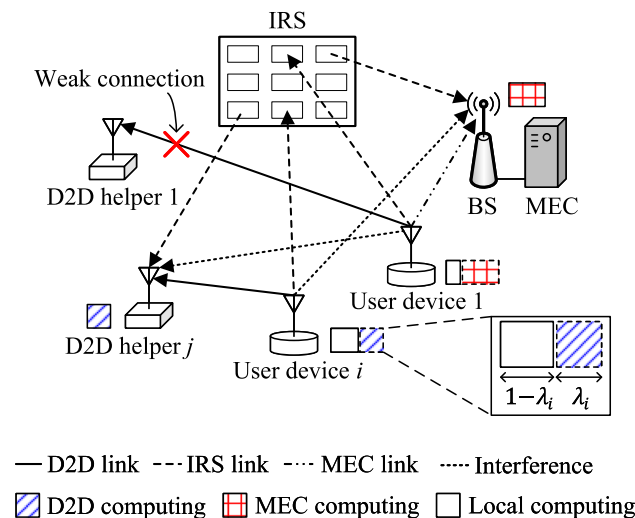


Fig. 1 IRS-enhanced D2D-MEC system model and offloading workflow. The system consists of a BS connected to a MEC server, multiple D2D user-helper pairs, and an IRS with M reflecting elements that assists both D2D and device-to-MEC communications. The diagram highlights local computing, D2D helper computing, MEC computing, direct and IRS-reflected links, interference among users, and the task offloading ratio λ_i (with $1 - \lambda_i$ computed locally)

in $\mathcal{N}_2$, so we have $N_1 = N_2 = N$. We further denote $\{0\}$ as the index for the MEC server.

2.1 Communication model

Here, we use “helper device” to refer to either a D2D helper device or the MEC server. Let $h_{ij} \in \mathbb{C}^{1 \times 1}$, $\mathbf{h}_{iI}^H \in \mathbb{C}^{1 \times M}$, and $\mathbf{H}_{Ij} \in \mathbb{C}^{M \times 1}$ denote, respectively, the channel between user device i and helper device j , the channel between user device i and the IRS, and the channel between the IRS and helper device j . The phase-shift of the IRS is denoted as $\Theta = \text{diag}(e^{j\theta_1}, e^{j\theta_2}, \dots, e^{j\theta_M})$.

In IRS-assisted offloading, the overall channel gain consists of the direct channel and the IRS-reflected link. Therefore, the effective channel between user device i and helper device j can be expressed as

$$h_{ij}^{\text{eff}} = h_{ij} + \mathbf{h}_{iI}^H \Theta \mathbf{H}_{Ij}. \quad (1)$$

When user device i offloads its tasks to helper device j , the transmission rate modeled as the Shannon capacity (Tian et al. 2025) is given by

$$r_{ij} = B \log_2 \left(1 + \frac{P_i |h_{ij}^{\text{eff}}|^2}{\sum_{k \in \mathcal{N}_1 / \{i\}} P_k |h_{kj}^{\text{eff}}|^2 + \sigma^2} \right), \quad i \in \mathcal{N}_1, j \in \mathcal{N}_2 + \{0\}, \quad (2)$$

where B is the bandwidth, P_i is the transmit power of user device i , and σ^2 denotes the noise power.

2.2 Local computing model

The computation task for user device i is defined as $Q_i = \{s_i, F_i, t_{i,\max}\}$, where s_i , F_i , and $t_{i,\max}$ are the task size [bits], the required CPU cycles to compute one bit [CPU cycles per bit], and the maximum tolerable latency [seconds], respectively. To simplify task computation, we set $t_{i,\max} = 1$ s, so we have $Q_i = \{s_i, F_i\}$.

The time required for local computing is given as

$$T_i^{\text{loc}} = x_i \frac{(1 - \lambda_i) s_i F_i}{f_i^{\text{loc}}}, \quad (3)$$

where $x_i \in \{0, 1\}$ is the binary indicator for local computing, λ_i is the task offloading ratio for user device i , and f_i^{loc} is the local computing capability of user device i . The energy consumption of local computing is expressed by

$$E_i^{\text{loc}} = P_i^{\text{loc}} T_i^{\text{loc}}, \quad (4)$$

where P_i^{loc} is the local processing power.

2.3 Partial offload computing model

2.3.1 D2D helper computing

When user device i offloads tasks to the D2D helper device j , its total offloading latency is the sum of transmission latency and processing latency, which can be denoted as

$$T_{ij}^{\text{D2D}} = y_i \lambda_i s_i \left(\frac{1}{r_{ij}} + \frac{F_i}{f_j^{\text{D2D}}} \right), \quad (5)$$

where f_j^{D2D} is the computing capability of D2D helper device j , and $y_i \in \{0, 1\}$ is a binary indicator that equals 1 if user device i users D2D helper computing and 0 otherwise. The energy consumption of user device i when offloading tasks to D2D helper device j is expressed as

$$E_{ij}^{\text{D2D}} = y_i \lambda_i s_i \left(\frac{P_i}{r_{ij}} + \frac{P_i^{\text{idle}} F_i}{f_j^{\text{D2D}}} \right), \quad (6)$$

where P_i^{idle} is the circuit power consumption of user device i .

2.3.2 MEC computing

When user device i offloads its tasks to the MEC server, the latency and energy consumption of user device i are

calculated as

$$T_{i0}^{\text{MEC}} = z_i \lambda_i s_i \left(\frac{1}{r_{i0}} + \frac{F_i}{f_i^{\text{MEC}}} \right), \quad (7)$$

$$E_{i0}^{\text{MEC}} = z_i \lambda_i s_i \left(\frac{P_i}{r_{i0}} + \frac{P_i^{\text{idle}} F_i}{f_i^{\text{MEC}}} \right), \quad (8)$$

where f_i^{MEC} is the CPU frequency at the MEC server assigned to the computing task of user device i , and $z_i \in \{0, 1\}$ is a binary indicator that equals 1 if device i offloads to the MEC server and 0 otherwise.

Each user device is assumed to offload its tasks to either a single D2D helper device or the MEC server at any given time ($y_i + z_i = 1$) (Dai et al. 2022). Then, the offloading latency for user device i is obtained by combining (5) and (7) as

$$\begin{aligned} T_i^{\text{off}} &= T_{ij}^{\text{D2D}} + T_{i0}^{\text{MEC}} \\ &= \lambda_i s_i \left[y_i \left(\frac{1}{r_{ij}} + \frac{F_i}{f_j^{\text{D2D}}} \right) + z_i \left(\frac{1}{r_{i0}} + \frac{F_i}{f_i^{\text{MEC}}} \right) \right]. \end{aligned} \quad (9)$$

Similarly, the offloading energy consumption for user device i is obtained by summing (6) and (8), given as

$$\begin{aligned} E_i^{\text{off}} &= E_{ij}^{\text{D2D}} + E_{i0}^{\text{MEC}} \\ &= \lambda_i s_i \left[y_i \left(\frac{P_i}{r_{ij}} + \frac{P_i^{\text{idle}} F_i}{f_j^{\text{D2D}}} \right) + z_i \left(\frac{P_i}{r_{i0}} + \frac{P_i^{\text{idle}} F_i}{f_i^{\text{MEC}}} \right) \right]. \end{aligned} \quad (10)$$

2.4 Problem formulation

In this section, we discuss the joint MEC offloading and resource allocation problem, aiming to maximize computing efficiency over latency. From the computing models in Sections 2.2 and 2.3, the total offloading latency and energy consumption of the user device i can be expressed as

$$T_i^{\text{tot}} = T_i^{\text{loc}} + T_i^{\text{off}}, \quad (11)$$

and

$$E_i^{\text{tot}} = E_i^{\text{loc}} + E_i^{\text{off}}, \quad (12)$$

respectively. We denote $\mathcal{I}_i = \{x_i, y_i, z_i\}$ as the computing mode for user device i . The total computed bits are defined as the sum of locally computed bits and offloaded bits. Accordingly, the computing efficiency of user device i , defined as

the ratio of total computed bits to energy consumption, is calculated as

$$CE_i^{\text{tot}} = \frac{x_i (f_i^{\text{loc}} / F_i) + y_i r_{ij} + z_i r_{i0}}{E_i^{\text{tot}}}. \quad (13)$$

From (11) and (13), we formulate the optimization model as

$$\max_{\mathcal{I}, f, \theta, \lambda, P} \sum_{i=1}^N \frac{CE_i^{\text{tot}}}{T_i^{\text{tot}}}, \quad (14)$$

subject to C1: $\{x_i, y_i, z_i\} \in \{0, 1\}$,

$$\text{C2: } x_i + y_i + z_i \geq 1,$$

$$\text{C3: } y_i + z_i \leq 1,$$

$$\text{C4: } T_i^{\text{tot}} < T_{\max},$$

$$\text{C5: } \sum f_i^{\text{MEC}} \leq f_{\max},$$

$$\text{C6: } P_i \leq P_{\max},$$

$$\text{C7: } 0 \leq \theta_m \leq 2\pi,$$

$$\text{C8: } 0 \leq \lambda_i \leq 1.$$

Here, C1 enforces that the decision variables are binary. C2 specifies the allowable task-placement options. The constraint of task computation can be only on the local device ($x_i = 1, y_i = 0, z_i = 0$), on the local device and a D2D helper ($x_i = 1, y_i = 1, z_i = 0$), on the local device and the MEC server ($x_i = 1, y_i = 0, z_i = 1$), or only on the MEC server ($x_i = 0, y_i = 0, z_i = 1$). C3 requires that a user device can only choose at most one helper, either a D2D helper or the MEC server. C4 limits the computing time to a predefined maximum time $T_{\max}$. C5 ensures that the total frequency provided to all devices from the MEC server does not exceed its maximum computing frequency. C6, C7, and C8 set the bounds for transmit power, IRS phase-shift, and offloading ratio, respectively.

3 Iterative methods

In this section, we explain two iteration-based algorithms, ES and GS with a barrier, for performance comparison against the proposed algorithm.

3.1 Exhaustive search

The ES algorithm (also called brute-force search) is a widely used approach for identifying the globally optimal solution to an optimization problem. Its implementation is straightforward and, when a solution exists, it guarantees discovery of the global-optimal solution by systematically evaluating

all feasible candidates. While this comprehensive approach ensures solution accuracy, it comes with the significant drawback of high computational costs. The computational cost of ES is proportional to the number of feasible candidates, which in many practical problems tends to grow very rapidly as the size of the problem and the quantization level increase.

In our work, the control parameters are $\mathcal{I}$, f , θ , λ , and p . The method entails a complete inspection of potential solutions to find which one maximizes the objective function while conforming to all constraint conditions.

3.2 Gradient search with barrier

As an iterative algorithm, gradient descent finds a local minimum of a differentiable function by moving in the direction of steepest descent (the negative gradient). We apply logarithmic barrier penalties (parameter $t > 0$) to handle constraints in GS. The logarithmic barrier function converts inequality constraints into penalty terms that provide regularization effects on the optimization process. This approach enables the algorithm to handle problems with difficult constraint sets and infeasible instances as effectively as feasible problems with simpler constraints, while reformulating the problem to one with only equality constraints (Curtis 2012).

In our work, we construct an alternative objective function by integrating this penalty function into the original objective function, which is defined as

$$U(\mathcal{I}, f, \theta, \lambda, P) = \sum_{i=1}^N \frac{CE_i^{\text{tot}}}{T_i^{\text{tot}}} + \omega, \quad (15)$$

where the penalty function is described by

$$\begin{aligned} \omega = \frac{1}{t} \sum_{i=1}^N \left[\ln(P_i) + \ln(\lambda_i) + \ln(f_i) + \ln(x_i) + \ln(y_i) \right. \\ \left. + \ln(z_i) + \ln(P_{\max} - P_i) + \ln(1 - \lambda_i) \right. \\ \left. + \ln(x_i + y_i + z_i - 1) + \ln(f_{\max} - \sum f_i) \right. \\ \left. + \ln(1 - y_i - z_i) + \sum_{m=1}^M (\ln(2\pi - \theta_m) + \ln(\theta_m)) \right]. \end{aligned} \quad (16)$$

The operational procedure of the GS with barriers is described in Algorithm 1. The $(\cdot)_{\text{old}}$ and $(\cdot)_{\text{new}}$ notations denote the variable at iteration t and $t + 1$, respectively. The $\eta_1, \eta_2, \eta_3, \eta_4$, and η_5 are the learning rates for each parameter.

Algorithm 1 Gradient Search Algorithm.

Initialize $\vec{\mathcal{I}}, \vec{f}, \vec{P}, \vec{\lambda}, \vec{\theta}$ in feasible region
 Set $\eta_1, \eta_2, \eta_3, \eta_4, \eta_5$ and ϵ
repeat
 $\vec{\mathcal{I}}_{\text{new}} = \vec{\mathcal{I}}_{\text{old}} + \eta_1 \times \frac{\partial U}{\partial (\vec{x}, \vec{y}, \vec{z})}$
 $\vec{f}_{\text{new}} = \vec{f}_{\text{old}} + \eta_2 \times \frac{\partial U}{\partial \vec{f}}$
 $\vec{P}_{\text{new}} = \vec{P}_{\text{old}} + \eta_3 \times \frac{\partial U}{\partial \vec{P}}$
 $\vec{\lambda}_{\text{new}} = \vec{\lambda}_{\text{old}} + \eta_4 \times \frac{\partial U}{\partial \vec{\lambda}}$
 $\vec{\theta}_{\text{new}} = \vec{\theta}_{\text{old}} + \eta_5 \times \frac{\partial U}{\partial \vec{\theta}}$
until $|U_{\text{new}} - U_{\text{old}}| \leq \epsilon$, with U_{new} and U_{old} are the loss functions at current and previous iteration, respectively.

4 Proposed DNN method

In this section, we propose a fully connected DNN structure, as depicted in Fig. 2. The proposed DNN takes the channel gain and the IRS phase-shift as inputs and outputs the transmit power of devices and the IRS phase-shift value.

First, we initialize the network weights with Xavier initialization, which speeds up convergence (Glorot and Bengio 2010). After setting the learning rate, the input data of the DNN is normalized using batch normalization, which results in zero mean and unit variance to help ensure convergence during the network training process (Ioffe and Szegedy 2015). The input data is given as

$$\vec{a} = \text{reshape} \left(\left(h_{ij}^{\text{eff}}, h_{iI}^H, h_{kj}, h_{kI}^H, H_{Li}, H_{Ik}, \Theta \right)^T \right), \quad (17)$$

$$\vec{x} = \mathcal{N}(\vec{a}), \quad (18)$$

where $\mathcal{N}(\cdot)$ is the normalization function. The input is reshaped from the matrix $\mathbf{A} \in \mathbb{R}^{I \times J}$ to a column vector $\vec{a} \in \mathbb{R}^Q$, where $Q = I \times J$. The number of neurons in the v -th hidden layer is set to $Q^{(v)}$, and $Q^{(0)}$ represents the number of inputs. The input consists of the channel of D2D user devices to BS, the channel of D2D user devices to IRS, the channel of IRS to BS, and the interference channel with the size of $12NM$.

Given an input vector, the output of neuron q at the v -th layer is computed as

$$y_q^{(v)} = g \left(\sum_{i=1}^{Q^{(v-1)}} w_{iq}^{(v-1)} y_i^{(v-1)} \right), \quad (19)$$

where $g(\cdot) = \frac{1}{1+e^{-\cdot}}$ is the sigmoid function, $w_{iq}^{(v-1)}$ is the weight connecting neuron i in the $(v-1)$ -th layer to neuron q in the v -th layer, $Q^{(v-1)}$ is the number of neurons at the

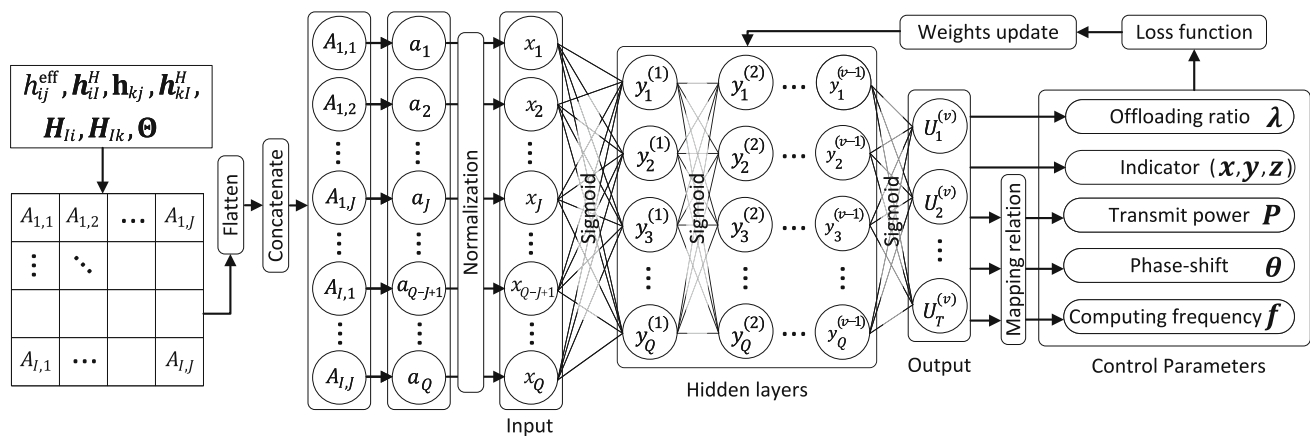


Fig. 2 Fully connected DNN architecture for IRS-assisted D2D-MEC partial offloading. The network inputs flattened, batch-normalized channel features for effective D2D-BS links, device-IRS links, IRS-BS links, interference channels, and the current IRS phase-shift matrix (θ), obtained by reshaping A into vector a and normalizing $\vec{x} = \mathcal{N}$. Hidden layers use sigmoid activation, and the output vector $U^{(v)} \in [0, 1]$ is mapped to control parameters, with a binary indicator I produced by

$(v-1)$ -th layer, and $y_q^{(0)} = x_q$. Each layer in the DNN is activated by the sigmoid function, which produces output values between zero and one. Therefore, it is necessary to map these outputs to control parameters whose ranges extend beyond $[0, 1]$. To provide valid control values, we use reversed min-max normalization to map the DNN outputs to the actual ranges of the target control parameters including transmit power, computing frequency, and phase-shift. This normalization technique has shown high precision across machine learning domains such as support vector machines and back-propagation neural networks (Sujon et al. 2024).

The mapping relations from the DNN output to transmit power, computing frequency, and phase-shift are given by

$$P_i = (1 - u_{[(12+M)i-(11+M)]}^{(v)}) P_{\min} + u_{[(12+M)i-(11+M)]}^{(v)} P_{\max}, \quad (20)$$

$$f_i = f_{\max} \times u_{[(12+M)i-(3+M)]}^{(v)}, \quad (21)$$

Algorithm 2 DNN-based D2D-MEC Partial Offloading.

Set the initial weights using the Xavier initialization (Glorot and Bengio 2010)

Calculate the DNN input using (17) and (18)

Set an Error Tolerance and learning rate ζ

repeat

 Compute the hidden layer outputs along with the DNN output using (19)

 Establish the connection between the DNN output and the normalized control parameters using (21) and (22)

 Compute the error function based on (24) and (26)

 Adjust the neural network weights using (25)

until $|\mathcal{L}_{\text{new}} - \mathcal{L}_{\text{old}}| \leq \text{Error Tolerance}$, with $\mathcal{L}_{\text{new}}$ and $\mathcal{L}_{\text{old}}$ are the loss functions at current and previous iteration, respectively.

thresholding. Training follows Algorithm 2 with Xavier initialization, batch normalization, forward propagation, output-to-control mapping, a Lagrange-dual loss that embeds the objective and inequality constraints, and weight updates until convergence. The figure depicts the data flow as Normalization $\rightarrow$ Flatten/Concatenate $\rightarrow$ Hidden layers (sigmoid) $\rightarrow$ Output/control mapping $\rightarrow$ Loss function $\rightarrow$ Weights update

$$\theta_k = 2\pi \times u_{[(12+M)i-(M-1)]:[(12+M)i]}^{(v)} \quad (22)$$

With $\vec{U}^{(v)}$ restricted to the range $[0, 1]$, the transmit power in (20) always satisfies the power constraint inequality. For the phase-shift of the IRS and the computing frequency, due to the phase range $\theta \in [0, 2\pi]$ and the computing frequency range $f \in [f_{\min}, f_{\max}]$, we multiply the output of the sigmoid function with 2π and $f_{\max}$ directly as in (21) and (22), respectively. For the parameter $\mathcal{I}$, we set $\mathcal{I} = \mathbb{I}(u_{[(12+M)i-(6:8+M)]}^{(v)} \geq 0.5)$, where $\mathbb{I}(\cdot)$ is the indicator function. It is noted that hard thresholding is non-differentiable and prevents gradient-based learning. To enable gradient flow despite the non-differentiable thresholding operation, we employ the clipped straight-through estimator (STE) (Bengio et al. 2013), which approximates the backward gradient as the identity function with clipping as $\frac{\partial I_i}{\partial u_i^{(v)}} \approx 1$ if $u_i^{(v)} \in (0, 1)$ and 0 otherwise. This allows joint optimization of continuous and discrete parameters while preventing gradient explosion. Other parameters such as $\lambda, \alpha, \beta, \gamma, \delta, \kappa$ and v are taken directly from the output because they have the same range as the sigmoid's $[0, 1]$ range. Therefore, the number of outputs from the proposed DNN is equal to the number of control parameters, $N_c \times N$.

We construct the loss function using a Lagrange dual form by subtracting the objective function and adding the inequality constraints, given as

$$\mathcal{L}(\mathcal{I}, f, \theta, \lambda, P, \alpha, \beta, \gamma, \delta, \kappa, v) = - \sum_{i=1}^N \frac{CE_i^{\text{tot}}}{T_i^{\text{tot}}} + \sum_{i=1}^N \left[\alpha_i (P_i - P_{\max}) + \delta_i (\lambda_i - 1) \right]$$

$$\begin{aligned}
& + \kappa_i (y_i + z_i - 1) + \nu_i (1 - (x_i + y_i + z_i)) \\
& + \gamma_i \left(\sum_{i=1}^N f_i^{\text{MEC}} - f_{\max} \right) \Big] + \sum_{m=1}^M \beta_m (\theta_m - 2\pi),
\end{aligned} \quad (23)$$

where $\alpha, \beta, \gamma, \delta, \kappa$ and ν are the Lagrange multipliers. The weight update rules are obtained by taking the derivative of the loss function with respect to the transmit power, IRS phase-shift, computing frequency, offloading ratio, offloading decisions, and the Lagrange multiplier parameters ($\alpha, \beta, \gamma, \delta, \kappa$ and ν), so both primal variables and dual variables are updated via backpropagation within one learning loop.

These derivatives are denoted as $\nabla_{\theta} \mathcal{L}, \nabla_P \mathcal{L}, \nabla_f \mathcal{L}, \nabla_{\lambda} \mathcal{L}, \nabla_x \mathcal{L}, \nabla_y \mathcal{L}$ and $\nabla_z \mathcal{L}$, respectively. The updates are expressed as

$$\begin{aligned}
\mu_{[(12+M)i-(K-1)]:[(12+M)i]}^{(V+1)} &= \nabla_{\theta} \mathcal{L}, \\
\mu_{[(11+M)i-(11+M)]}^{(V+1)} &= \nabla_P \mathcal{L}, \\
\mu_{[(12+M)i-(10+M)]}^{(V+1)} &= \nabla_f \mathcal{L}, \\
\mu_{[(12+M)i-(9+M)]}^{(V+1)} &= \nabla_{\lambda} \mathcal{L}, \\
\mu_{[(12+M)i-(8+M)]}^{(V+1)} &= \nabla_x \mathcal{L}, \\
\mu_{[(12+M)i-(7+M)]}^{(V+1)} &= \nabla_y \mathcal{L}, \\
\mu_{[(12+M)i-(6+M)]}^{(V+1)} &= \nabla_z \mathcal{L}, \\
\mu_{[(12+M)i-(5+M)]}^{(V+1)} &= P_i - P_{\max}, \\
\mu_{[(12+M)i-(4+M)]}^{(V+1)} &= \theta_i - 2\pi, \\
\mu_{[(12+M)i-(3+M)]}^{(V+1)} &= \sum_{i=1}^N f_i^{\text{MEC}} - f_{\max}, \\
\mu_{[(12+M)i-(2+M)]}^{(V+1)} &= \lambda_i - 1, \\
\mu_{[(12+M)i-(1+M)]}^{(V+1)} &= y_i + z_i - 1, \\
\mu_{[(12+M)i-M]}^{(V+1)} &= 1 - (x_i + y_i + z_i).
\end{aligned} \quad (24)$$

The weight of the v -th layer is updated by

$$W_{iq}^{(v)} \leftarrow W_{iq}^{(v)} - \zeta y_q^{(v-1)} \mu_q^{(v)}, \quad (25)$$

where ζ represents the learning rate extending from the v -th hidden layer to the input layer. Hence, the error function corresponding v -th hidden layer is formulated as

$$\mu_i^{(v)} = y_i^{(v)} \left(1 - y_i^{(v)} \right) \sum_{q=1}^{Q^{(v+1)}} W_{iq}^{(v+1)} \mu_q^{(v+1)}. \quad (26)$$

Here, we set the number of layers of the proposed DNN to 4 layers, and the number of neuron equal to output size of

the DNN. The optimal weights in the proposed DNN are obtained when the error tolerance is satisfied, as shown in Algorithm 2.

It is noted that variable bounds are handled cleanly from the above update procedures. Box constraints for phase $\theta \in [0, 2\pi]$, computing frequency $f \in [f_{\min}, f_{\max}]$, and other $[0, 1]$ -bounded parameters are enforced by sigmoid mappings and scaling, while remaining coupling or resource constraints, including transmit power, phase upper bound, MEC frequency budget, and decision-sum constraints, are enforced through the Lagrange dual penalty terms in the loss function. For the discrete or indicator-type decisions, thresholding the network outputs (e.g., $I = 1\{\cdot \geq 0.5\}$) provides binary behavior, and the dual terms still penalize violations of associated feasibility relations (e.g., $x_i + y_i + z_i \geq 1$), which integrate mixed-integer structure into the training objective. Practically, the Lagrange multipliers act as adaptive penalties and reduce the need for manual penalty tuning and encourage satisfaction of constraints as training proceeds.

5 Simulation results and complexity analysis

This section presents the numerical results obtained using the proposed DNN method, the fixed-offloading DNN, ES, GS, and REINFORCE (Thomas and Brunskill 2017). The “fixed-offloading DNN” refers to a DNN model where all tasks are fully offloaded to the edge for computation. In this approach, the parameters are fixed ($x_i = 0, y_i = 0, z_i = 1$ and $\lambda = 1$), while the remaining control parameters are optimized using the proposed DNN. The REINFORCE algorithm is a classic policy gradient method that estimates expected cumulative reward through Monte Carlo sampling. It iteratively generates trajectories to produce unbiased gradient estimates, using the cumulative rewards to define the loss function for policy updates. A comprehensive list of simulation parameters and their corresponding values is provided in Table 1. During the simulation, task sizes are randomly generated within a predefined range between the minimum and maximum task sizes. Training parameters are configured with 0.008 for the learning rate, 8,192 for the mini-batch size, and 0.1 for dropout regularization. The training data generation process begins with the random initialization of key system parameters to create diverse channel realizations. User device locations are randomly distributed within the considered coverage area of the BS, ensuring a wide variety of spatial configurations. Similarly, the IRS phase-shifts are randomly generated for each of the M reflecting elements, with values uniformly sampled from the feasible range $[0, 2\pi]$. Based on these randomly initialized positions and phase-shifts, the corresponding channel gains are computed, including the direct channels between user devices and helper devices (h_{ij}), the channels from user devices to

Table 1 Summary of key simulation parameters for IRS-assisted D2D-MEC

Parameter	Symbol	Value
Number of D2D pair	N	2
Bandwidth	B	5 MHz
Min., Max. transmit power	$P_{\min}, P_{\max}$	0dBm, 20dBm
Minimum task size	$s_{\min}$	0.1 Mbits
Maximum task size	$s_{\max}$	[0.5, 1, ..., 3] Mbits
Maximum delay	$T_{\max}$	5s
Local device power	p_i^{loc}	[0.1-0.5] W
Circuit power	p_i^{idle}	[0.0001-0.01] W
Local computing resources	f_i^{loc}	$[0.5-2] \times 10^9$ cycle/s
Required CPU cycles/bit	F_i	[500-2000] cycle/bit
Edge computing resources	$f_{\max}$	$[10-90] \times 10^9$ cycle/s
DNN learning rate	ζ	0.001
GS learning rate	$\{\eta_1, \eta_3, \eta_4, \eta_5\}, \eta_2$	0.0001, 0.001

the IRS ($\mathbf{h}_{IJ}^H$), and the channels from the IRS to helper devices ($\mathbf{H}_{IJ}$). Through this randomization process, the DNN models are trained on 5×10^5 randomly generated independent channel realizations, and their performance is evaluated using 10^4 additional random channel samples. All algorithms are implemented using MATLAB and executed on a computer with an Intel Core i5-1340P processor, an NVIDIA GeForce RTX 2070 graphics card, and 32 GB RAM.

Figure 3 presents the convergence behavior of the proposed DNN evaluated across four different learning rates of 0.002, 0.005, 0.008, and 0.015 over 250 training iterations. We also provide additional comparison against various

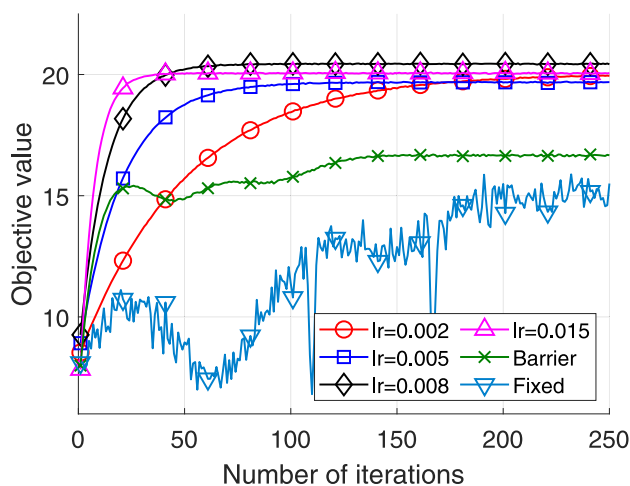
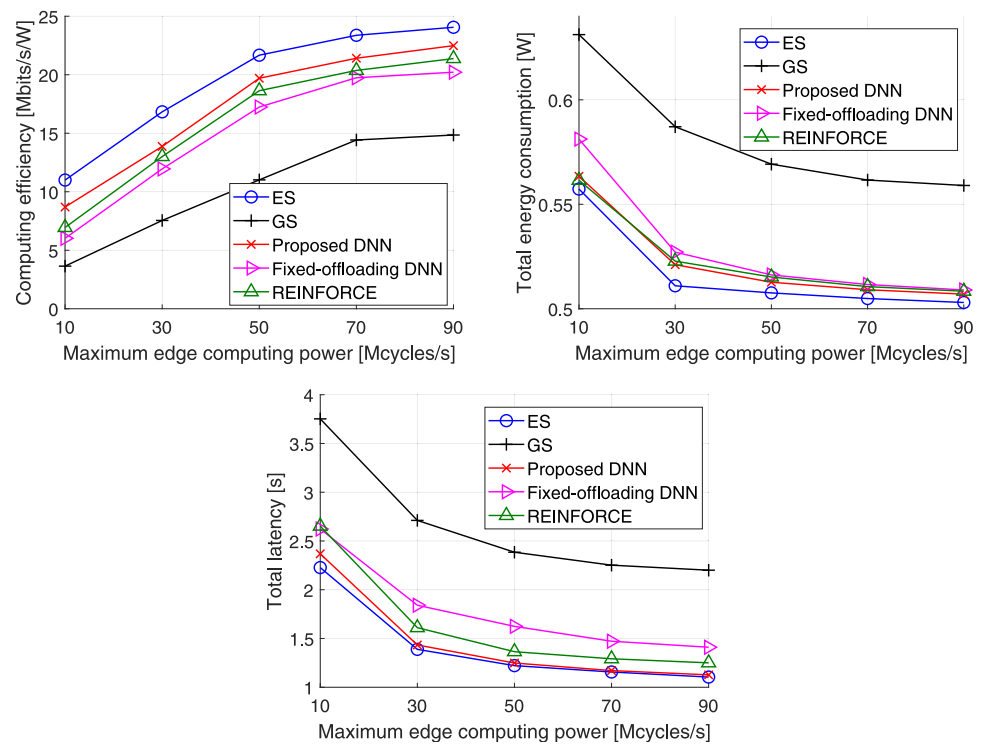


Fig. 3 Objective value versus iteration number for Lagrange-dual method with varying learning rates ($lr=0.002, 0.005, 0.008, 0.015$), barrier method, and fixed penalty method. The Lagrange-dual approaches converge smoothly to optimal values (19–20) within 100–150 iterations, with 0.008 learning rates providing highest convergence value. The barrier method (Barrier) converges to a suboptimal value (around 17), while the fixed penalty method (Fixed) shows severe oscillations and poor convergence throughout 250 iterations

loss functions, i.e., Logarithmic barrier loss and fixed-valued Lagrangian loss (hereafter referred to as Barrier and Fixed, respectively), using the same DNN configuration as the proposed DNN. It is observed that Barrier performs slightly higher than the Fixed, and the learning process is less noisy than Fixed. This is due to the logarithmic barrier's inherently smooth, self-regulating penalty mechanism that dynamically adjusts gradient magnitudes based on proximity to constraint boundaries, whereas fixed Lagrangian multipliers impose static penalties that cannot adapt to varying constraint violation intensity, resulting in abrupt gradient changes and oscillatory updates when constraints transition between satisfied and violated states during training. For various learning rate of the proposed DNN, the results clearly demonstrate the significant impact of learning rate selection on convergence characteristics. Higher learning rates of 0.015 and 0.008 achieve rapid convergence, but the learning rate of 0.015 achieves a slightly lower objective value than that of 0.008. This is because the higher learning rate can overshoot the optimum and cause oscillations around the optimal solution. The learning rate of 0.005 exhibits intermediate convergence behavior and gradually saturates at around 19.5 after approximately 150 iterations. In contrast, the lowest learning rate of 0.002 displays the slowest convergence pattern and shows continuous but gradual improvement throughout the entire training period. Overall, we can verified that the proposed DNN using Lagrange-dual function with various learning rates outperform Fixed and Barrier.

In Fig. 4 (a), computing efficiency (Mbits/s/W) consistently improves with increasing maximum edge computing power. The proposed DNN achieves very close computing efficiency to ES, followed by the *REINFORCE*, fixed-offloading DNN and GS. This explains that fully offloading tasks to be computed at the edge is not optimal even higher edge computing power is available, whereas the proposed

Fig. 4 Impact of maximum edge computing power on (a) computing efficiency, (b) total energy consumption, and (c) total latency for ES, GS, proposed DNN, Fixed-offloading DNN, and REINFORCE. The trend displays enhanced performance for all algorithms as the maximum edge computing power increases

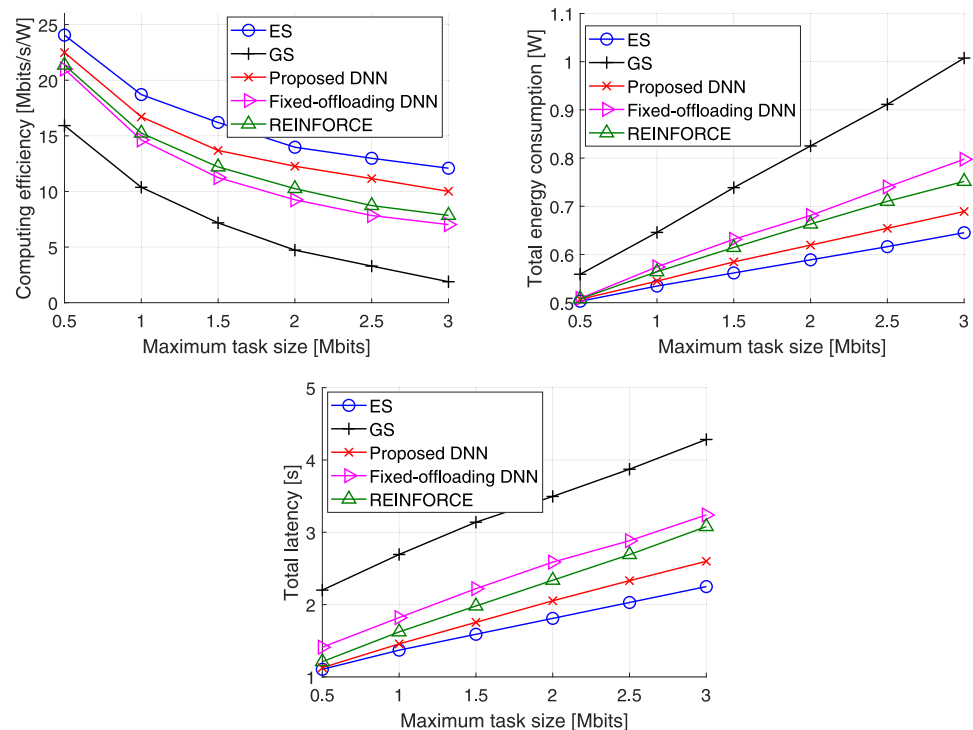


DNN benefits from opportunistic task offloading by keeping small tasks local or on nearby D2D devices and sending only the most demanding tasks to the edge. Figure 4 (b) shows total energy consumption against the maximum edge computing power. As the edge's available computing frequency increases, the total energy required by the system decreases across all algorithms because more computation can be offloaded to the MEC server, which results in reducing the task transmission of devices and local CPU energy. *It is also noticed that REINFORCE achieves total energy consumption similar to the proposed DNN.* Figure 4 (c) shows total latency as a function of the maximum edge computing power. The proposed DNN outperforms other algorithms due to its unique algorithmic characteristics. Unlike REINFORCE, which suffers from high variance due to Monte Carlo sampling for policy gradient estimation, the DNN employs deterministic backpropagation with stable gradient computation. The embedded Lagrange-dual loss function simultaneously optimizes primal variables (offloading decisions, partial offloading ratio, edge computing frequency, transmit power, and IRS phase shifts) and dual variables (Lagrange multipliers) in a unified learning loop that enables coherent constraint satisfaction without sampling inefficiency. While GS becomes trapped in local minima within the non-convex search space of coupled mixed-integer constraints, the DNN's deep architecture with batch normalization and Xavier initialization navigates the complex solution space more effectively. The sigmoid activations naturally enforce box constraints, and the STE enables

gradient flow through discrete thresholding operations. This achieves joint mixed-integer optimization that conventional gradient methods cannot handle. The fixed-offloading DNN, despite sharing the same architecture to the proposed DNN, performs suboptimally by eliminating partial offloading flexibility ($\lambda = 1$), which prevents adaptive load balancing and resulting in 15-25% higher energy consumption. Increasing the edge computing capacity consistently reduces end-to-end latency for all schemes, as tasks can be processed more quickly at the MEC server than under constrained local resources, which results in shorter overall computation time.

Figure 5 (a) shows that computing efficiency decreases as the maximum task size increases from 0.5 to 3 Mbits, because larger payloads require more data transmission and computation, which increases energy consumption across all schemes. The proposed DNN closely tracks ES and remains superior to REINFORCE, fixed-offloading DNN, and GS over the evaluated range. In Fig. 5 (b), the relationship between the maximum task size (Mbits) and the total energy consumption (W) is verified. As the task size grows from 0.5 to 3 Mbits, the energy consumption also increases steadily. This figure shows that the proposed DNN maintains energy consumption levels as low close to those of ES, followed by REINFORCE. In contrast, GS, the fixed-offloading DNN, and REINFORCE consume significantly more energy compared to the proposed DNN. This is because larger task sizes require transferring more data between the local devices and D2D helper devices, whether it is via D2D or MEC server. This data transfer consumes energy that does not directly contribute to

Fig. 5 Impact of maximum task size on (a) computing efficiency, (b) total energy consumption, and (c) total latency for ES, GS, proposed DNN, Fixed-offloading DNN, and REINFORCE. The trend illustrates degraded performance for all algorithms as the maximum task size increases

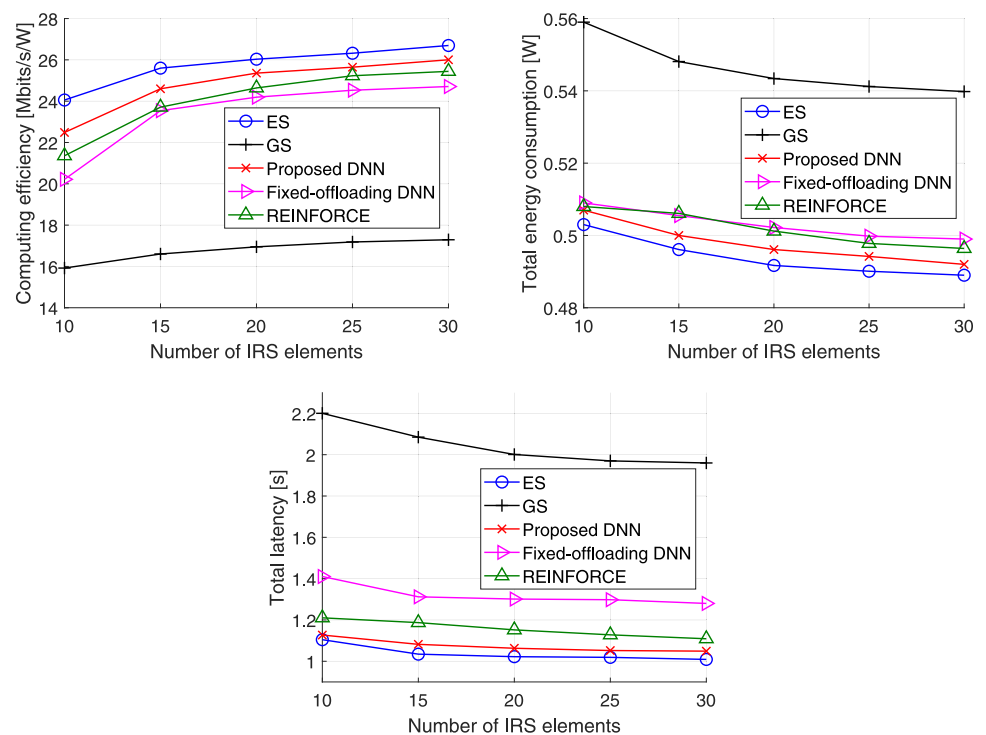


computation. In Fig. 5 (c), we compare the performance of the total latency versus the maximum task size. It is observed that the proposed DNN method achieves a near-global optimal solution as the maximum task size increases. The latency of the fixed-offloading DNN is significantly lower compared to GS. Additionally, REINFORCE achieves intermediate performance between the proposed DNN and fixed-offloading DNN due to fundamental algorithmic differences. While REINFORCE can adaptively optimize partial offloading and resource allocation unlike the rigid fixed approach, its episodic trajectory sampling requires extensive exploration across the high-dimensional action space. Moreover, REINFORCE's stochastic policy introduces action variability that compromises precision in fine-tuning continuous parameters (partial offloading ratio, edge computing frequency, transmit power, and IRS phase shifts), whereas the DNN's deterministic mapping ensures consistent solutions through direct constraint-encoded gradients.

Figure 6 (a) presents the increase in computing efficiency as the number of IRS elements increases. We can see that the proposed DNN achieves a near-optimal solution, especially once the IRS size reaches around 20 elements, due to improved signal quality. With higher SINR, data can be transmitted with lower power while maintaining the same reliability, which results in more bits per Watt. In addition, REINFORCE obtains solution closer to the proposed DNN, only slightly lower, when the number of IRS elements increase from 25 to 30. This trend can also be similarly observed at total latency and total energy consumption in

Fig. 6 (b) and (c), respectively. For GS, the computing efficiency remains very low for the entire evaluated range. The proposed DNN's higher performance across varying IRS elements is caused by its ability to learn the mapping between channel conditions and optimal control parameters through unsupervised training on diverse scenarios. Unlike GS, which performs independent optimization for each problem instance and repeatedly encounters the same local minima due to the non-convex coupled structure of IRS phase shifts and resource allocation, the DNN generalizes learned optimization patterns across different configurations, effectively applying learned resource allocation strategies to unseen channel realizations during inference. The Lagrange-dual loss function enforces constraint satisfaction during training by adaptively penalizing violations through dual variable updates that enables the network to internalize feasible solution structures and avoid infeasible regions. In contrast, GS must navigate constraint boundaries through iterative gradient projections that frequently overshoot or stagnate at suboptimal solutions where the gradient magnitude diminishes. As IRS elements increase, the DNN leverages its learned representations to efficiently handle the expanded phase-shift optimization, while GS struggles with the increased coupling between more IRS phases and other control variables, and REINFORCE requires significantly more trajectory samples to adequately explore the enlarged action space. Figure 6 (b) shows the total energy consumption versus the number of IRS elements. It is clear that energy consumption decreases as the number of IRS

Fig. 6 Impact of the number of IRS elements on (a) computing efficiency, (b) total energy consumption, and (c) total latency for ES, GS, proposed DNN, Fixed-offloading DNN, and REINFORCE. The trend shows improvement for all algorithms when the number of IRS elements increases



elements increases for all methods. The proposed DNN remains close to ES, while REINFORCE coincides with the fixed-offloading DNN for 10 to 20 IRS elements and shows slight improvement from 25 to 30 IRS elements. GS exhibits significantly higher total energy consumption across the entire evaluated range. Figure 6 (c) plots total latency versus the number of IRS elements and shows that latency decreases as the IRS size grows for all schemes. This reduction results from improved link quality as more IRS elements increase the effective SINR, which provides higher data rates and thus shortens offloading and processing times. Across the range, the proposed DNN achieves latency very close to ES and consistently outperforms REINFORCE, GS and the fixed-offloading DNN, which shows that IRS-assisted partial D2D/MEC offloading effectively reduces delay.

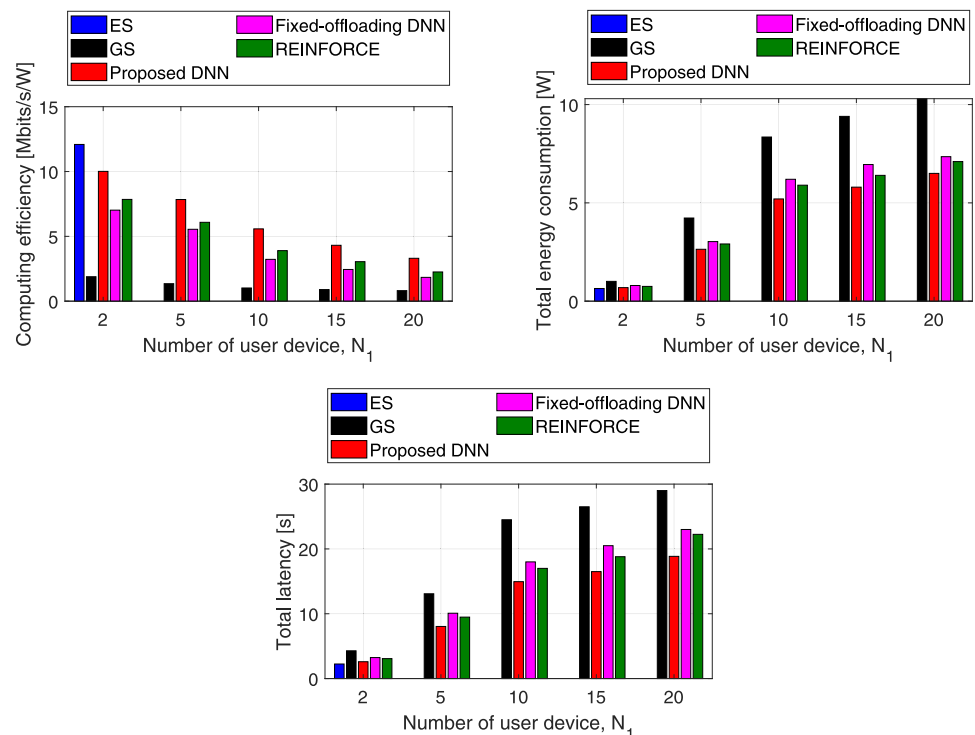
Figure 7 evaluates system scalability by varying the number of user devices N_1 while examining computing efficiency, total energy consumption, and total latency. The number of helper devices is fixed at $N_2 = 5$ to assess performance under resource-constrained conditions. Due to the high computational complexity of ES, obtaining solutions becomes infeasible as N_1 increases (only provided at $N_1 = 2$); therefore, we focus our comparative analysis on the more scalable GS, fixed-offloading DNN, and REINFORCE algorithms.

Figure 7 (a) shows a declining trend as the number of user devices N_1 increases from 2 to 20. This reflects the inherent trade-off between resource utilization and system scalability. The proposed DNN algorithm achieves near-optimal performance and closely tracks the ES, with maximum deviation

of approximately 8% occurring at $N_1 = 2$. Notably, the proposed method substantially outperforms the GS and demonstrates efficiency gains exceeding 50% at larger user counts. The fixed-offloading DNN and REINFORCE approaches exhibit inferior performance due to their inability to dynamically adapt partial offloading ratios. The convergence of REINFORCE toward DNN performance at higher user densities suggests that reinforcement learning-based approaches require sufficient problem complexity to fully exploit their exploratory mechanisms (Hao et al. 2023). In contrast, the proposed DNN training with embedded Lagrange-dual constraints enables better convergence characteristics even under constrained optimization scenarios.

Figure 7 (b) describes the total energy consumption versus the number of user devices. Total system energy consumption increases monotonically with increasing user devices. It transitions from approximately 1.2 W at $N_1 = 2$ to approximately 7.0 W at $N_1 = 20$. This reflects accumulated transmission power and circuit energy dissipation across multiple simultaneous offloading operations. The proposed DNN method, followed by REINFORCE, maintains energy consumption comparable to ES, with relative performance gap remaining within 3–5% throughout the evaluated range. This demonstrates robust optimization despite increased problem dimensionality. The GS exhibits severe performance degradation and consumes approximately 40–50% more energy than the DNN approach at $N_1 = 20$. This indicates fundamental limitations in GS for handling coupled resource allocation constraints in large-scale systems.

Fig. 7 Impact of the number of user devices on (a) computing efficiency, (b) total energy consumption, and (c) total latency for ES, GS, proposed DNN, Fixed-offloading DNN, and REINFORCE. The pattern shows diminished performance for all algorithms as the number of user devices increases



The fixed-offloading baseline unconditionally offloads all tasks to the MEC server regardless of local computing availability. It consistently consumes 15–25% more energy than the proposed partial offloading strategy and validates the critical importance of adaptive task partitioning in energy-constrained edge computing environments.

Figure 7 (c) illustrates the total latency as a function of the number of user devices. Total latency exhibits linear growth with increasing user device count. It rises from approximately 2.8 s at $N_1 = 2$ to approximately 22 s at $N_1 = 20$. This characteristic indicates delays that accumulate at both wireless communication links and MEC server processing systems. The proposed DNN achieves latency performance within 5–10% of the ES reference across various number of user devices. This demonstrates better constraint satisfaction and computational efficiency compared to other approaches. The GS exhibits particularly poor scaling behavior and reaches approximately 29 s at $N_1 = 20$. This represents a 32% increase relative to the DNN approach and underscores the inadequacy of GS for handling mixed-integer resource allocation problems under tightly coupled system constraints. The REINFORCE achieves latency performance comparable to or slightly below the DNN at higher user densities. This suggests that model-free learning approaches may offer comparable benefits in highly dense multi-user scenarios, albeit at substantially increased computational overhead during both training and inference phases.

To sum up, from Figs. 4, 5, 6, and 7, the proposed DNN method achieves near-global-optimal solutions comparable

to ES, whereas REINFORCE exhibits marginally lower performance than the proposed DNN, followed by GS with substantial performance degradation. Moreover, the proposed DNN method outperforms fixed-offloading DNN, which shows that partial D2D and MEC offloading effectively reduce both energy consumption and latency of users.

Table 2 shows the computational complexity analysis of the proposed DNN, GS and ES. It shows that both the computational complexities of the proposed DNN and GS are much lower than ES. However, the number of operations of the proposed DNN is significantly lower than those of GS and ES. The computational complexity of the proposed DNN is evaluated during the inference stage, where the trained DNN approximates the optimal control variables ($\mathcal{I}, f, \theta, \lambda, P$), which is $O(12N_c N^2 M s)$. The DNN has the most manageable computational complexity among comparable algorithms, scaling polynomially with network size (the number of inputs and outputs). Based on the analyses (Lee et al. 2021, 2019), the computation time is typically on the order of milliseconds, which allows the proposed DNN to support real-time operations. The computational cost of GS is directly determined by the convergence error threshold ω (Boyd and Vandenberghe 2004), as a higher threshold (higher error) requires less computational time, but results in reduced solution precision. Conversely, a lower threshold may lead to a more accurate solution, but with a trade-off in more computational time. ES scales with the quantization rate Q_s , N , and M . As the quantization rate Q_s increases, this also raises the computational cost, because a higher Q_s requires

Table 2 Computational complexity and operation count comparison ($N = 2$, $M = 10$, $s = 3$, $\omega = 10^{-4}$, $Q_s = 50$)

Algorithm	Computational complexity	Number of operation
Proposed DNN	$O(12N_c N^2 Ms)$	6.3×10^4
GS	$O(\omega^{-2})$	1×10^8
ES	$O((2Q_s)^{N+M})$	1×10^{30}
REINFORCE	$O(\iota^{-4})$	1×10^{16}

Asymptotic complexity and actual number of operations for the proposed DNN, GS, ES, and REINFORCE. The proposed DNN achieves the lowest computational cost (6.3×10^4 operations), demonstrating significant advantages over GS (10^8), REINFORCE (10^{16}), and ES (10^{30})

more granular evaluations of possible solutions. *The REINFORCE algorithm, based on policy gradient methods, incurs substantial computational overhead with $O(\iota^{-4})$ complexity and 1×10^{16} operations. This high computational cost stems from its requirement for multiple episode rollouts to estimate policy gradients and the inverse quartic dependence on variance tolerance that necessitates extensive sampling to achieve stable convergence.*

6 Conclusion

6.1 Summary

This work improves computing efficiency and latency in IRS-enhanced D2D-MEC systems, where computational tasks can be offloaded to either D2D devices or an MEC server. The optimization problem is formulated with both linear and non-linear constraints, considering the control parameters including offloading decisions, offloading ratio, edge computing frequency, transmit power, and IRS phase-shift. The proposed DNN integrates Xavier initialization, batch normalization, and sigmoid activations, and uses a Lagrange-dual loss to embed both the maximization objective (computing efficiency over latency) and the system's inequality constraints. Outputs are mapped to feasible transmit powers, computing frequencies, and IRS phase-shifts, with both primal and dual variables updated via backpropagation to enforce box and coupling constraints during training. Simulations across varying maximum task sizes, edge computing capacities, and IRS element counts show that the DNN consistently attains solutions close to ES and outperforms the GS, REINFORCE algorithm, and fixed-offloading methods in computing efficiency, total energy consumption, and end-to-end latency. Comparative evaluation of different loss functions demonstrates that the proposed Lagrange-dual approach achieves superior convergence speed and solution quality compared to fixed penalty and logarithmic barrier methods. Convergence analysis confirms rapid and stable training with appropriate learning-rate selection, and results highlight that partial D2D/MEC offloading is superior to full offloading even under ample edge resources.

Increasing the number of IRS elements improves effective SINR, and achieves higher data rates, lower energy, and reduced latency, compared to comparison methods. The REINFORCE algorithm, while demonstrating comparable energy efficiency, exhibits higher latency due to gradient variance inherent in policy gradient methods, particularly in scenarios with limited problem complexity. From a computational perspective, the DNN's inference complexity scales polynomially with network size ($O(12N_c N^2 Ms)$), which achieves near-global optimal solutions with low computational complexity compared to ES, and significantly lower than the $O(\iota^{-4})$ complexity of REINFORCE that necessitates extensive episode rollouts for stable convergence.

6.2 Limitations and future work

Future works will address several important limitations of the current framework and extend it toward more realistic and practical scenarios as follows. Regarding imperfect CSI, in real-world wireless systems, perfect channel state information is rarely available due to estimation errors, feedback delays, or quantization effects, and these imperfections can significantly degrade the performance of IRS-assisted communication and optimization algorithms. Future research will focus on developing robust optimization and learning strategies that explicitly account for uncertainty in CSI, as understanding how CSI accuracy affects performance is essential for practical IRS-assisted D2D-MEC implementation. Concerning user mobility and time-varying channels, dynamic user movement introduces rapidly changing channel conditions, which can invalidate static optimization solutions. To maintain reliable performance, the existing framework will incorporate adaptive or more robust learning mechanisms that respond to mobility, such as online learning, transfer learning, or predictive channel modeling, while extensions could also explore the trade-off between adaptation speed and computational complexity to ensure tracking fast-varying channels without excessive overhead. For practical deployment and real-world validation, while simulations provide valuable insights, real-world experimentation is essential to understand the impact of hardware limitations, environmental propagation effects, and coordination challenges in

IRS-assisted D2D-MEC scenarios, as field trials can help identify unexpected constraints such as IRS alignment errors, limited phase-shift resolution, or interference from surrounding devices.

By addressing these challenges, future work can bridge the gap between theoretical design and practical implementation, enabling IRS-assisted D2D-MEC networks to achieve reliable, adaptive, and high-performance operation in real-world conditions.

Author Contributions Kimcheang Chhea: Investigation, Software, Validation, Formal analysis, Writing, Methodology, Visualization, Investigation. Jung-Ryun Lee: Conceptualization, Methodology, Validation, Writing — review & editing, Funding acquisition, Project administration, Supervision.

Funding This work was supported in part by the Ministry of Science and ICT (MSIT), Korea through Information Technology Research Center (ITRC) Support Program under Grant IITP-RS-2025-00436248 Supervised by the Institute for Information & Communications Technology Planning & Evaluation(IITP), and in part by the National Research Foundation of Korea (NRF) Grant funded by the Korean Government (MEST) under Grant RS-2023-00251105.

Declarations

Competing interests The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

References

- Almasaeid HM (2025) Uav-assisted mobile edge computing model for cognitive radio-based iot networks. *Computer Communications* p 108071
- Bengio Y, Léonard N, Courville A (2013) Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*
- Boyd SP, Vandenberghe L (2004) *Convex optimization*. Cambridge University Press, Cambridge
- Cao X, Wang F, Xu J et al (2018) Joint computation and communication cooperation for energy-efficient mobile edge computing. *IEEE Internet Things J* 6(3):4188–4200
- Chai R, Lin J, Chen M et al (2019) Task execution cost minimization-based joint computation offloading and resource allocation for cellular d2d mec systems. *IEEE Syst J* 13(4):4110–4121
- Chai Zx, Chai Zy, Ren J et al (2025) Computation offloading and task caching in the cloud–edge collaborative iovs: A multi-objective evolutionary algorithm. *Simulation Modelling Practice and Theory* p 103087
- Chen Z, Zhang Z, Yang Z (2024) Big ai models for 6g wireless networks: Opportunities, challenges, and research directions. *IEEE Wireless Communications*
- Curtis FE (2012) A penalty-interior-point algorithm for nonlinear constrained optimization. *Math Program Comput* 4(2):181–209
- Dai X, Xiao Z, Jiang H et al (2022) Task co-offloading for d2d-assisted mobile edge computing in industrial internet of things. *IEEE Trans Industr Inf* 19(1):480–490
- Fereira RJ, Ranaweera C, Lee K et al (2025) Energy efficient resource management for real-time iot applications. *Internet of Things* p 101515
- Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics, JMLR Workshop and Conference Proceedings*, pp 249–256
- Hao J, Yang T, Tang H et al (2023) Exploration in deep reinforcement learning: From single-agent to multiagent domain. *IEEE Transactions on Neural Networks and Learning Systems* 35(7):8762–8782
- He Y, Ren J, Yu G et al (2019) D2d communications meet mobile edge computing for enhanced computation capacity in cellular networks. *IEEE Trans Wireless Commun* 18(3):1750–1763
- Ioffe S, Szegedy C (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: *International conference on machine learning, pmlr*, pp 448–456
- Jin B, Xu X (2024) Forecasts of thermal coal prices through gaussian process regressions. *Ironmaking & Steelmaking* 51(8):819–834
- Jin B, Xu X (2024) Gaussian process regression based silver price forecasts. *Journal of Uncertain Systems* 17(03):2450013
- Jin B, Xu X (2025a) China commodity price index (ccpi) forecasting via the neural network. *International Journal of Financial Engineering* pp 1–27
- Jin B, Xu X (2025b) Contemporaneous causal orderings among prices of retail properties: evidence from chinese cities through vector error-correction modeling and directed acyclic graphs. *Journal of Financial Management of Property and Construction* 30(ahead-of-print)
- Jin B, Xu X, Zhang Y (2025) Peanut oil price change forecasts through the neural network. *foresight* 27(3):595–612
- Lee H, Lee SH, Quek TQ (2019) Deep learning for distributed optimization: Applications to wireless resource management. *IEEE J Sel Areas Commun* 37(10):2251–2266
- Lee W, Lee K, Choi HH et al (2021) Deep learning for swipt: Optimization of transmit-harvest-respond in wireless-powered interference channel. *IEEE Trans Wireless Commun* 20(8):5018–5033
- Li C, Xia J, Liu F et al (2021) Dynamic offloading for multiuser multi-cap mec networks: A deep reinforcement learning approach. *IEEE Trans Veh Technol* 70(3):2922–2927
- Li D (2022) Bound analysis of number configuration for reflecting elements in irs-assisted d2d communications. *IEEE Wireless Communications Letters* 11(10):2220–2224
- Liu Y, Hu Q, Cai Y et al (2021) Latency minimization in intelligent reflecting surface assisted d2d offloading systems. *IEEE Commun Lett* 25(9):3046–3050
- Mao S, Chu X, Wu Q et al (2021) Intelligent reflecting surface enhanced d2d cooperative computing. *IEEE Wireless Communications Letters* 10(7):1419–1423
- Peng J, Qiu H, Cai J et al (2021) D2d-assisted multi-user cooperative partial offloading, transmission scheduling and computation allocating for mec. *IEEE Trans Wireless Commun* 20(8):4858–4873

- Seo YJ, Lee J, Hwang J et al (2021) A novel joint mobile cache and power management scheme for energy-efficient mobile augmented reality service in mobile edge computing. *IEEE Wireless Communications Letters* 10(5):1061–1065
- Spinelli F, Mancuso V (2020) Toward enabled industrial verticals in 5g: A survey on mec-based approaches to provisioning and flexibility. *IEEE Communications Surveys & Tutorials* 23(1):596–630
- Sujon KM, Hassan RB, Towshi ZT et al (2024) When to use standardization and normalization: empirical evidence from machine learning models and xai. *IEEE access*
- Sun Y, Xia H, Su C et al (2025) A multi-agent enhanced ddpg method for federated learning resource allocation in iot. *Computer Communications* p 108066
- Thomas PS, Brunskill E (2017) Policy gradient methods for reinforcement learning with function approximation and action-dependent baselines. *arXiv preprint arXiv:1706.06643*
- Tian B, Xin X, Zhang Q et al (2024) Mec-enabled edge network deployment with converged fiber and millimeter-wave communications. *IEEE Transactions on Communications*
- Tian S, Xiang S, Zhou Z et al (2025) Task offloading and resource allocation based on reinforcement learning and load balancing in vehicular networking. *IEEE Transactions on Consumer Electronics*
- Van NTT, Nguyen HT, Luong NC et al (2022) Intelligence reflecting surface-aided integrated data and energy networking coexisting d2d communications. *IEEE Trans Wireless Commun* 21(11):10035–10049
- Wang S, Zhao S, Gui H et al (2025) Energy-efficient collaborative task offloading in multi-access edge computing based on deep reinforcement learning. *Ad Hoc Netw* 169:103743
- Wang W, Yang L, Meng A et al (2021) Resource allocation for 5g-aided 5g-comp downlink cellular networks with underlaying d2d communications. *IEEE Trans Wireless Commun* 21(6):4295–4309
- Xu X (2020) Corn cash price forecasting. *Am J Agr Econ* 102(4):1297–1320
- Xu X, Zhang Y (2021) Individual time series and composite forecasting of the chinese stock index. *Machine Learning with Applications* 5:100035
- Xu X, Zhang Y (2022) Contemporaneous causality among one hundred chinese cities. *Empirical Economics* 63(4):2315–2329
- Yang D, Zhan C, Yang Y, et al (2025) Integrating uavs and d2d communication for mec network: A collaborative approach to caching and computation. *IEEE Transactions on Vehicular Technology*
- Yazici İ, Shayea I, Din J (2023) A survey of applications of artificial intelligence and machine learning in future mobile networks-enabled systems. *Eng Sci Technol Int J* 44:101455

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.