Digital Twin-Assisted URLLC-Enabled Task Offloading in Mobile Edge Network via Robust Combinatorial Optimization

Yixue Hao, *Member, IEEE*, Jiaxi Wang, Dongkun Huo, Nadra Guizani, Long Hu, and Min Chen, *Fellow, IEEE*

***Abstract*— Digital twin (DT)-assisted mobile edge network can achieve energy-efficient task offloading by optimizing the decision-making in real time. Although many DT-assisted task offloading solutions in mobile edge networks have been designed, stochastic asynchronizations between the DTs and physical entities are still ignored. In this paper, we investigate a task offloading problem in a DT-assisted URLLC-enabled mobile edge network which considered the uncertain deviation between DT estimated values and physical actual values. Specifically, we formulate a latency and energy consumption minimization problem by optimizing task offloading, resource allocation, and power management. To solve this problem, we propose a DT-assisted robust task offloading scheme (DTRTO) based on learning composed of decision and deviation networks. The deviation network predicts the worst-case deviations based on the pre-decision, and the decision network optimize the decision considered the worst-case deviation. The simulation results show that, compared to the baseline algorithms, the DTRTO scheme can realize low latency and energy consumption in task offloading while maintaining high robustness.**

***Index Terms*— Digital twin, mobile edge network, robust combinatorial optimization.**

I. Introduction

THE sixth generation (6G) networks integrates emerging technological advances in wireless communications, to provide extremely high transmission rates and low latency, satisfying promising services and applications. With the advent of the 6G era, hundreds of billions of mobile devices will be connected to wireless networks owing to the flexible, convenient, and ubiquitous intelligent services provided by mobile networks [1]. However, smart services require intensive computation tasks and persistent processing with low latency, posing a challenge for mobile devices with limited computing resources. Mobile edge computing (MEC) [2] reduces end-to-end latency by offloading computing tasks to surrounding base stations equipped with powerful computing resources. Moreover, the massive amount of offloading tasks also requires low latency and highly reliable communication transmissions. For example, virtual reality health [3] is required to transmit monitored body conditions of the patient continuously in real time, and personalized diagnosis provided by deep learning [4]. Thus, the low latency requirement is a major challenge for 6G systems in providing high-quality services for emerging intelligent applications.

The ultra-reliable low latency communication (URLLC) [5] technology has emerged as a viable solution because it establishes performance requirements for emerging latency-sensitive applications in 6G networks, which are achieved through the transmission of shorter data packets. Recently, the URLLC-enabled mobile edge network has attracted much attention. However, given the high-performance demands and limited computing and transmission resources of the URLLC-enabled mobile edge network, the challenges of designing an optimal task offloading scheme with efficient resource allocation increase significantly owing to the dynamics of the networks.

Digital twin (DT) for mobile edge network attempts to provide a comprehensive functional description for mobile edge networks to reproduce the attributes, behaviors, and rules of physical entities and directly optimize physical operations via a real-time feedback control. Thus, DT has attracted the attention from researchers and is expected to assist in operational phases to optimize the decision-making of task offloading and resource allocation in the URLLC-enabled mobile edge networks. For example, a DT scheme of URLLC-enabled mobile edge network was presented in [6] to address the optimal problem of latency and reliability by optimizing various communication and computation variables. The Lyapunov optimization method and actor-critic deep reinforcement learning are leveraged in [7] to minimize the offloading latency under the constraints of accumulated consumed service migration cost during user mobility in DT edge networks. To solve a practical end-to-end latency minimization problem in the DT-assisted mobile edge network, Tan et al. [8]

Manuscript received 1 December 2022; revised 20 May 2023; accepted 3 August 2023. Date of publication 30 August 2023; date of current version 26 October 2023. This work was supported by National Natural Science Foundation of China (NSFC) under Grant 62276109, Grant 62176101, and Grant 61821003. The work of Yixue Hao was supported by the Hubei Natural Science Foundation under Grant 2021CFB050. *(Corresponding author: Min Chen.)*

Yixue Hao, Jiaxi Wang, Dongkun Huo, and Long Hu are with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: yixuehao@hust.edu.cn; jiaxiwang@hust.edu.cn; dongkunhuo@hust.edu.cn; hulong@hust.edu.cn).

Nadra Guizani is with the Department of Computer Science and Engineering, The University of Texas at Arlington, Arlington, TX 76010 USA (e-mail: nadra.guizani@uta.edu).

Min Chen is with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510640, China, and also with the Pazhou Laboratory, Guangzhou 510640, China (e-mail: minchen@ieee.org).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JSAC.2023.3310051.

Digital Object Identifier 10.1109/JSAC.2023.3310051

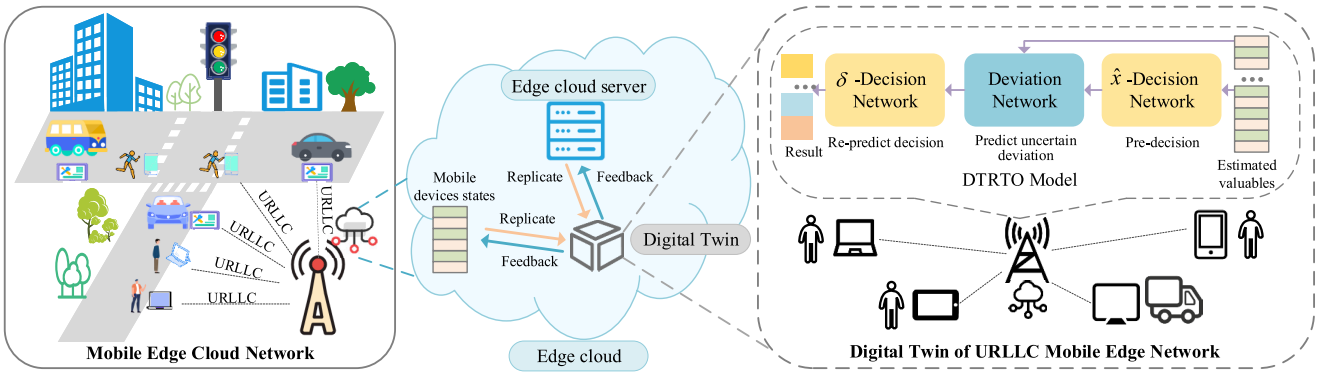


Fig. 1. Digital Twin-assisted URLLC-enabled mobile edge network architecture.

iteratively optimize the transmission power of Internet-of-Things (IoT) devices, user association, intelligent task offloading, and estimated CPU processing rate of the devices. These studies have effectively improved the quality of service of mobile edge networks.

Considering the DT-assisted mobile edge network, the URLLC model plays a significant role in ensuring ultra-reliable and low latency transmission between DTs and physical entities. To achieve such a novel scenario, it is urgent to reduce the latency and energy consumption of task offloading and resource allocation. However, the existing DT-assisted mobile edge networks focus only on the design and implementation of the network architecture, ignoring the stochastic asynchronization between the DT and the physical entities, such as the data deviations of task request and position of mobile devices, especially ignoring that these deviations are dynamically changing. Huynh et al. [6] considered the deviation of processing frequency, but they designed the algorithm by fixing the deviation. In fact, the deviation is uncertain and difficult to predict. Whether the constructed models and generated decisions can effectively guide the real network is an uncertain issue. Moreover, the task offloading problems are a difficult problem to solve, especially when considering uncertain variables. Massive optimization iteration and complex processing are required, leading to long decision-making latency, but DT is expected to provide feedback on the optimistic result in real time.

To address these problems, in this paper, we focus on the uncertain deviation in DT-assisted URLLC-enabled mobile edge networks and propose an effective DT-assisted robust task offloading scheme (DTRTO) based on learning to minimize the latency and energy consumption. Specifically, as shown in Fig. 1, we consider a task offloading problem with uncertain deviation and take the URLLC and DT into the mobile edge network to realize the high Quality of Experience (QoE). In the physical environment, various types of mobile devices transmit tasks requests to the edge cloud server deployed on the base station using URLLC. In the edge cloud server, twin models of physical entities are constructed. The DTRTO is deployed in the DT to predict the DT synchronization deviations, make the corresponding decisions, and send the decisions to the physical entities on time.

In summary, the main contributions of this paper include:

- Uncertain deviation: Considering the task offloading in the DT-assisted URLLC-enabled mobile edge network, we model the uncertain deviation between the physical entities and DT when DT monitors the network and makes the task offloading decision in real time.
- DTRTO scheme: We formulate a robust optimization problem and propose a novel DTRTO scheme based on learning to minimize latency and energy consumption with uncertain variables by optimizing the task offloading, resource allocation, and power management.
- Extensive performance evaluation: To evaluate the DTRTO scheme, we conduct extensive experiments. The experimental results show that, compared to the other algorithms, the DTRTO can significantly reduce task offloading latency and energy consumption with uncertain deviation.

The remainder of this paper is organized as follows. In Section II, we introduce the related works of task offloading for MEC and DT-assisted edge computing. In Section III, we formulate a task offloading problem on minimizing latency and energy consumption. In Section IV, we describe the composition and algorithms of the proposed DTRTO. The experimental comparisons are presented in Section V. Finally, the conclusion is given in Section VI.

II. Related Work

In this section, we introduce the related work from the following two aspects: task offloading for MEC and DT-assisted edge computing.

A. Task Offloading for Mobile Edge Computing

The MEC is one of the key technologies of 6G [9], which can greatly reduce system latency by providing powerful computing resources at the edge of the network. The MEC has been extensively studied for edge intelligence tasks, including analysis, prediction, and optimization.

As a fundamental technology in the MEC, computation offloading compensates for the shortcomings of mobile devices in terms of storage resources, computing capacity, and energy

efficiency [10]. On one hand, computation offloading of task requests not only relieves the communication pressure on the core networks, but also reduces the delay caused by long-distance data transmissions [11]. On the other hand, emerging applications in 6G also rely on the computation offloading technology for efficient service provisioning to users [12]. The computing and bandwidth capacity of edge servers and wireless links between edge servers to users are both limited.

Service providers prefer to hand over tasks to the edge cloud to improve the QoE of users due to a lower transmission delay. Yan et al. [13] propose a price-based distributed method to manage the offloaded tasks where each task can be arbitrarily divided bitwise for partial offloading. Zhang et al. [14] consider the heterogeneous energy harvesting MEC systems with multiple mobile devices and multiple edge servers. The problem of multi-hop multi-task partial offloading has been studied in [15] by considering the partial offloading among edge devices. Li et al. [16] propose an incentive-based offloading control framework for the MEC networks.

However, due to the dynamics of the networks, the obstacles in creating effective task offloading schemes considerably increase with high-performance demand with limited computing and transmission resources in mobile edge networks.

TABLE I
THE SUMMARY TABLE OF IMPORTATION NOTATIONS

Notation	Meaning
$\mathcal{M}$	The set of the mobile devices
DT_m^u	The digital twin model of the mobile device m
DT^e	The digital twin model of the edge cloud
s_m	The data size of offloading task of the mobile device m
c_m	The CPU cycle for unit data of offloading task of the mobile device m
d_m	The distance from the mobile device m to the edge cloud
$\hat{s}_m$, $\hat{c}_m$, $\hat{d}_m$	The DT estimated value of the task data size, the CPU cycle for unit data, and the distance from the mobile device m to the edge cloud
δ_s, δ_c, δ_d	The deviation between the real state and the estimated value
ϵ_s, ϵ_c, ϵ_d	The maximum range of the deviations
F_e	The maximum computing frequency provided by the edge cloud
f_m^e	The allocated computing frequency to the mobile device m
f_m^{loc}	The computing frequency of the mobile user m
p_m^{max}	The maximum transmission power of the mobile device m
p_m	The transmission power of the mobile device m
p_m^{min}	The minimum transmission power of the mobile device m
β_m	The task offloading decision of the mobile device m
R_m	The transmission rate of the mobile device m
h_m	The channel coefficient of the transmission channel between the mobile device m and the edge cloud
$\gamma(p_m, \omega_m)$	The SINR of the mobile device m
V_m	The channel dispersion of the transmission channel between the mobile device m and the edge cloud

B. Digital Twin Assisted Edge Computing

In recent years, DT technology has attracted widespread attention in academia and industry [17]. It is a digital copy of a physical entity and maps real physical entities and environments to a virtual space in real time [18]. Using intelligent learning algorithms and combined with the DT, the real-time data can help physical entities make more accurate and timely offloading decisions, reducing users' resource consumption [19]. The integration of DT and MEC [20] provides new possibilities for alleviating the resource constraints of DT applications in 6G networks.

As an exact replica of the physical system, the DT builds a bridge between the physical system and the digital space [21], [22]. DT technology, as one of the most critical enabling technologies in 6G communication, has been the subject of many advanced studies on improving the performance of MEC systems in edge networks [7]. Sun et al. [23] utilized DTs to capture the dynamic state of the network, and used unmanned aerial vehicle driven joint learning to realize the ground-air network. Lu et al. [24] proposed a digital double-edge network model, and developed a communication-efficient federated learning to improve the operating efficiency of the model. Li et al. [25] first proposes a double deep Q network algorithm derived from a deep reinforcement learning to effectively solve the problem of maximum transmission unit association and mobile device trajectory. Then, closed-form expressions are used to deal with the transmission power allocation problem, and the computational power allocation problem is further solved by an iterative algorithm.

Moreover, several studies have explored the use of DT technology as an important auxiliary tool to improve the performance of MEC systems. DTs are mostly used to carry out large-scale computing tasks to optimize the resource allocation and execution decisions for MEC and improve service quality. However, only few studies have focused on the data asynchronization and deviation between the DT network and the real network, ignoring the small differences between them [6], [7], [8], [23], such as the deviation of transmission rate and calculation frequency, etc. Huynh et al. considered the deviation of processing frequency, but they design the algorithm by fixing the deviation. In fact, the deviation is uncertain and difficult to predict [6]. Thus, we focus on the uncertain deviation between physical entities and DT. Based on the real data collected by the physical entities, we train the agent model to predict the deviation, improve the robustness of the decisions, and guide the physical entities to make appropriate resource allocations.

III. System Model and Problem Formulation

In this section, we consider a task offloading problem in a DT-assisted URLLC-enabled mobile edge network and formulate an optimal problem for minimizing latency and energy consumption.

A. System Overview

The system model is shown in Fig 2, which contains the physical layer, the DT layer, and the URLLC model. It aims to solve the problem of how to optimize the task offloading, resource allocation, and power management for mobile devices in the DT-assisted URLLC-enabled edge network with uncertain deviations of physical entities and DTs. In the physical world, mobile devices transmit tasks to the edge cloud by the URLLC. On the edge cloud server, the DT communicate with each other through the intra twin and the DT communicate

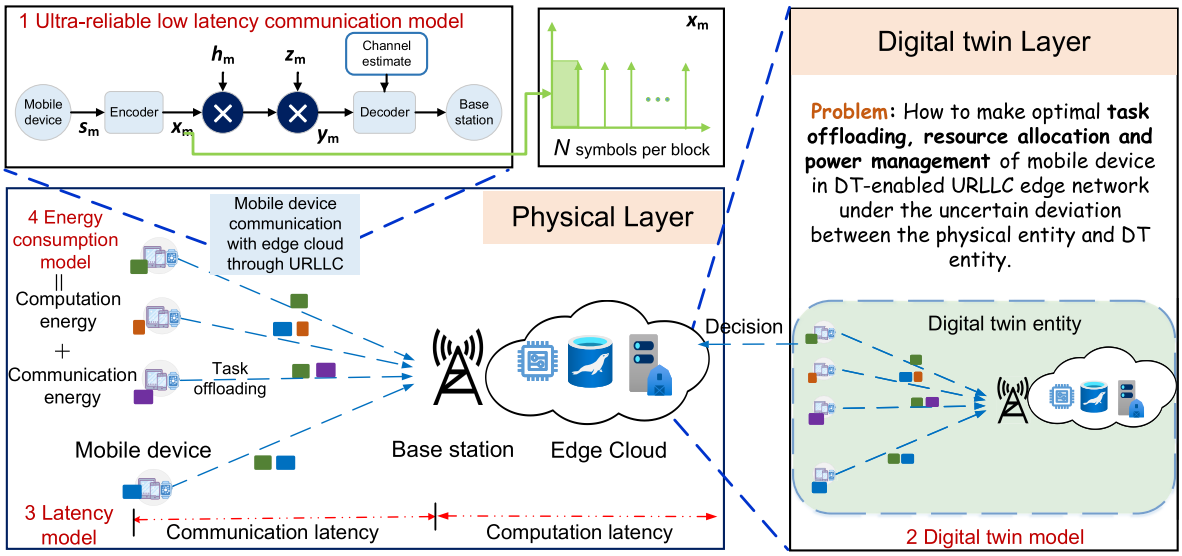


Fig. 2. An overview of the system model.

with the physical base station with the help of inter twin. The DT makes the optimal decision and sends the decision result to the physical entity in time.

In the physical layer, there is one edge cloud and M mobile devices. We denote $\mathcal{M} = \{1, 2, \cdots, m, \cdots M\}$ as the set of mobile devices, they request tasks for processing at the edge cloud through URLLC. The task generated by the mobile device m is represented by $Q_m = \{s_m, c_m\}$. We denote the data size of the computing task as s_m in [bits] and the required CPU cycles for one-bit data as c_m in [cycles/bit] which is related to the task complexity. Let $\beta_m \in [0, 1]$ be the ratio of the m-th offloaded task, where β_m of the task is computed on the edge cloud completely while $(1 - \beta_m)$ of the task is computed on the local device. The main notations adopted in this work are listed in Table I.

B. Communication Model

There are M mobile devices and one base station, each device is equipped with a single antenna, and the base station is equipped with L antennas. The mobile devices move in a way that obeys the random direction movement model. The channel vector between the m-th device and the base station, denoted by $h_m \in \mathbb{C}^{L\times 1}$, can be modeled as $h_m = \sqrt{g_m}\bar{h}_m$. The g_m is the large-scale channel coefficient and can be expressed as $g_m = 10^{PL(d_m)/10}$, where $PL(d_m)$ is the passloss and d_m is the distance from the m-th mobile device to the edge cloud. The $\bar{h}_m$ is the small-scale fading which follows the Rayleigh fading model $\bar{h}_m \sim \mathcal{CN}(0, I_L)$. Under a shared wireless medium, the $L \times 1$ received signal vector at the base station can be expressed as

$$y = \sum_{m\in\mathcal{M}} h_m\sqrt{p_m}x_m + z_m \tag{1}$$

where p_m and x_m are the transmit power and unit-power data symbol of the m-th device. $z_m \sim \mathcal{CN}(0, I_L)$ is the additive white Gaussian noise (AWGN) with zero mean and unit variance.

We define the estimated channels as $\hat{H} = [\hat{h}_1, \hat{h}_2, \ldots, \hat{h}_M]$ and channel estimation errors as $\tilde{H} = [\tilde{h}_1, \tilde{h}_2, \ldots, \tilde{h}_M]$. Let A be an $\mathcal{L} \times \mathcal{M}$ linear detection matrix that is based on the estimated channel $\hat{H}$. By using the linear detection A, the received signal can be processed as

$$y^D = A^H y \tag{2}$$

In this paper, the base station treat the estimated channel as the true channel, denote the a_m as the m-th column of matrix A, the SINR for the m-th device γ_m is given by

$$\gamma(p_m, \Omega_m) = \Omega_m p_m |a_m^H \hat{h}_m|^2 / (\sum_{i\in\mathcal{M}\setminus m} \Omega_i p_i |a_m^H \hat{h}_i|^2 + \sum_{i\in\mathcal{M}} \Omega_i p_i |a_m^H \tilde{h}_i|^2 + ||a_m||^2) \tag{3}$$

where $\Omega_m \in \{0, 1\}$, and $\Omega_m = 1$ (or 0) means if the channel from m-th mobile device to the base station is connected (or not). By transmitting short packets, the wireless URLLC systems can satisfy the stringent latency constraints. In this paper, we assume that both devices and the base station have perfect channel state information. The work in [26] gave the conclusion that with a fixed block length N and a block error probability not larger than ϵ_u, the maximum coding rate (in bits per channel use) of using an error-correcting code, can be expressed as

$$R_{\epsilon_u} = C_m - \sqrt{\frac{V_m}{N}} Q^{-1}(\epsilon_u) + O(\frac{\log N}{N}) \tag{4}$$

where C_m is the channel capacity, and V_m denotes the channel dispersion between the m-th device and the base station. V_m measures the stochastic variability of the channel relative to a deterministic channel with the same capacity [27]. It can be expressed as

$$V_m = 1 - (1 + \gamma_m)^{-2} \tag{5}$$

$Q^{-1}(\cdot)$ represents the inverse Q-function, which can be expressed as $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{t^2}{2}}$, and B_m is the bandwidth

of the m-th device. We denote $\omega = \sum_{m\in\mathcal{M}} \Omega_m/N$. Then the uplink URLLC coding rate in the finite block length regime is obtained by:

$$R_m(p_m,\Omega_m) \approx (1-\omega)B_m \log\left(1+\gamma(p_m,\Omega_m)\right) - B_m\sqrt{\frac{(1-\omega)V_m(p_m,\Omega_m)}{N}}Q^{-1}(\epsilon_u) \quad (6)$$

C. Digital Twin Model

The DT layer is placed on the edge cloud which replicates the mobile devices, and edge cloud via a real-time updating including time-variant state information, historical data, offloading strategy, etc. By synchronizing the overall situation including the mobile device position, task requests, and wireless transmission capabilities, the DT layer decides the optimal offloading strategies for the physical entity layer. The DT consists of one edge cloud duplicate and M mobile device duplicates, so it can be expressed as

$$DT = (DT^e, DT_1^u, DT_2^u, \cdots, DT_M^u) \quad (7)$$

The DT duplicate of the edge cloud DT_e is denoted as

$$DT^e = \Phi(F_e) \quad (8)$$

where F_e is the maximum computing frequency of the edge cloud. The m-th mobile device duplicate DT_m^u can be expressed as

$$DT_m^u = \Phi(\hat{s}_m, \hat{c}_m, \hat{d}_m, p_m^{\max}) \quad (9)$$

where $\hat{s}_m$, $\hat{c}_m$ and $\hat{d}_m$ are the estimated task data size, CPU cycle for unit data, and distance from mobile device to the edge cloud respectively, $p_m^{\max}$ is the maximum transmission power of the m-th mobile device.

The DT duplicates the state of the physical entity layer over time. Because the DT layer is placed on the edge server, it is relatively easy to obtain the most accurate estimated state of the edge device. However, the location and task requests of the mobile devices change constantly, so there may be errors between the DT estimated value and the mobile entities. Such uncertain and unpredictable errors between the physical entities and the DT affect the system performance. Therefore, we consider the twin model of the m-th mobile device with an uncertain deviation δ, and the real task data size, the CPU cycle for unit data, and the distance can be expressed as

$$\begin{cases} s_m = \hat{s}_m + \delta_s^m, & |\delta_s| \le \epsilon_s \\ c_m = \hat{c}_m + \delta_c^m, & |\delta_c| \le \epsilon_c \\ d_m = \hat{d}_m + \delta_d^m, & |\delta_d| \le \epsilon_d \end{cases} \quad (10)$$

where δ_s, δ_c, and δ_d represent the deviation between the real value and the estimated value, and ϵ_s, ϵ_c and ϵ_d are the maximum range of deviations.

D. System Latency Model

The communication contains the uplink for transmitting the task data from mobile devices to the edge cloud and the downlink for sending feedback results from the edge cloud to mobile devices. But the downlink communication latency is much smaller than the uplink communication latency because usually, the data size of the feedback results is much smaller than the task data. So we only consider the uplink communication latency in the system latency model. We can get the uplink transmission rate R_m of the m-th device from (6). The transmission latency can be expressed as

$$T_m^{tr}(\beta_m, s_m, d_m, \delta^m) = \frac{\beta_m s_m}{R_m} \quad (11)$$

Then, the latency for computing task of the m-th mobile device can be expressed as

$$T_m^{pr}(\beta_m, c_m, s_m, \delta^m) = \frac{\beta_m s_m c_m}{f_m^e} + \frac{(1-\beta_m)s_m c_m}{f_m^{loc}} \quad (12)$$

where f_m^e is the edge cloud computing frequency providing to the mobile device m and f_m^{loc} is the computing frequency of the m-th mobile device. The allocated computing resources cannot exceed the maximum computing frequency of the edge cloud

$$\sum_{m=1}^{M} f_m^e \le F_e \quad (13)$$

The average latency of all tasks is given by

$$T_{avg} = \frac{1}{M}\sum_{m=1}^{M}(T_m^{tr} + T_m^{pr}) \quad (14)$$

E. Energy Consumption Model

The energy consumption model of mobile device is divided into transmission and computing energy consumptions. The transmission energy consumption can be expressed as

$$E_m^{tr}(\beta_m, p_m) = \beta_m p_m T_m^{tr} \quad (15)$$

For the m-th mobile device, its transmission power should be less than the maximum transmission power p_m^{max} and more than the received transmission power of base station $p_m^{\min}$.

$$p_m^{\min} \le p_m \le p_m^{\max} \quad (16)$$

Then, the computing energy consumption can be expressed as

$$E_m^{pr}(\beta_m, c_m) = k\beta_m c_m (f_m^{loc})^2 \quad (17)$$

where f_m^{loc} is the computing frequency of the m-th mobile device and k is the unit computing energy consumption. So the average energy consumption is expressed as

$$E_{avg} = \frac{1}{M}\sum_{m=1}^{M}(E_m^{tr} + E_m^{pr}) \quad (18)$$

F. Problem Formulation

Due to the limited computing and transmission resources, the DT decides the task offloading resource allocation, and power management strategy for the optimal system performance with an uncertain deviation. We denote the resources decision provided to mobile devices as

$$\begin{cases} \beta = \{\beta_1, \beta_2, \cdots, \beta_M\} \\ \mathbf{f^e} = \{f_1^e, f_2^e, \cdots, f_M^e\} \\ \mathbf{p} = \{p_1, p_2, \cdots, p_M\} \end{cases} \quad (19)$$

Then, we denote the estimated valuables of mobile devices as

$$\begin{cases} \hat{\mathbf{s}} = \{\hat{s}_1, \hat{s}_2, \cdots, \hat{s}_M\} \\ \hat{\mathbf{c}} = \{\hat{c}_1, \hat{c}_2, \cdots, \hat{c}_M\} \\ \hat{\mathbf{d}} = \{\hat{d}_1, \hat{d}_2, \cdots, \hat{d}_M\} \end{cases} \tag{20}$$

Moreover, we denoted the uncertain deviation between estimated and true of mobile device as

$$\begin{cases} \delta_s = \{\delta_s^1, \delta_s^2, \cdots, \delta_s^M\} \\ \delta_c = \{\delta_c^1, \delta_c^2, \cdots, \delta_c^M\} \\ \delta_d = \{\delta_d^1, \delta_d^2, \cdots, \delta_d^M\} \end{cases} \tag{21}$$

Thus, the objective of the DT-assisted URLLC edge network is to minimize the latency and energy consumption of the physical systems assisted by the DT. To avoid the unevenness caused by the magnitude values of the latency or energy consumption, we adjust the impact factors α_1 and α_2 according to the actual situations. Formally, the optimization function can be expressed as

$$\mathcal{P}_1 : \min_{\beta, f^e, p} \max_{\delta_s, \delta_c, \delta_d} \alpha_1 T_{avg} + \alpha_2 E_{avg} \tag{22}$$

$$\text{subject to } C1 : \beta_m \in [0,1], \forall m \in \mathcal{M} \tag{23}$$

$$C2 : \sum_{m=1}^{M} f_m \leq F_e \tag{24}$$

$$C3 : p_m^{\min} \leq p_m \leq p_m^{\max}, \forall m \in \mathcal{M} \tag{25}$$

$$C4 : |\delta| \leq \epsilon \tag{26}$$

where δ is the uncertain deviation between the physical entity and the DT. Constraint $C1$ means the offloading decision. Constraint $C2$ represents the limitation of the computing capability of the edge cloud and constraint $C3$ represents the transmission power limitation of the mobile devices. Constraint $C4$ means that δ cannot exceed the range of the uncertain set ϵ.

Problem $\mathcal{P}_1$ is a min-max problem with uncertain variables which is hard to solve in general. One effective approach is to use a traditional robust optimization method. However, the conventional maximizer needs to be executed once for each sampled decision in order to select the optimal robust decision and the uncertain set is difficult to denote. Thus, we propose a robust optimization network based on learning to solve the optimal decision with uncertain deviation.

IV. Learning to Robust Optimize

In this section, we propose a learning-based algorithm DTRTO based on a robust combinatorial optimization. Specifically, we first decompose the optimization problem $\mathcal{P}_1$ into two sub-problems, i) the minimization problem corresponding to the optimization of task offloading, resource allocation and power management, and ii) the maximization problem corresponding to uncertain deviation. Then, we design two novel learning networks to solve the subproblems.

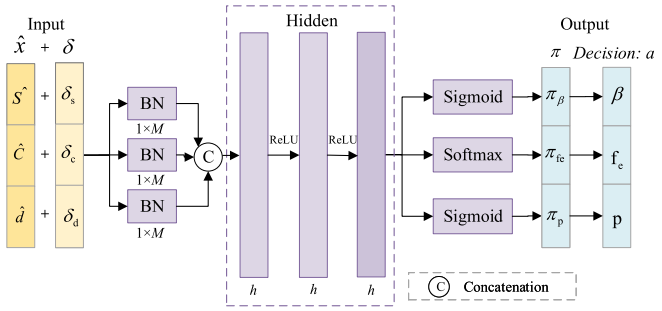


Fig. 3. Decision network for DT-assisted task offloading.

A. Decision Network for DT-Assisted Task Offloading

First, according to (19) and (20), we define the estimated variable as $\hat{x} = \{\hat{s}, \hat{\mathbf{c}}, \hat{\mathbf{d}}\}$ and the decision variable $\mathcal{M}$ as $a = \{\beta, \mathbf{f^e}, \mathbf{p}\}$. Then the minimization problem can be reformulated as

$$\mathcal{P}_2 : \min_{\beta, f^e, p} \alpha_1 T_{avg}(\hat{x}, a) + \alpha_2 E_{avg}(\hat{x}, a) \tag{27}$$

$$\text{subject to } C1 : \beta_m \in [0,1], \forall m \in \mathcal{M} \tag{28}$$

$$C2 : \sum_{m=1}^{M} f_m \leq F_e \tag{29}$$

$$C3 : p_m^{\min} \leq p_m \leq p_m^{\max}, \forall m \in \mathcal{M} \tag{30}$$

We denote the optimization objective function as

$$\mathcal{F} = \alpha_1 T_{avg} + \alpha_2 E_{avg} \tag{31}$$

The object $\mathcal{P}_2$ is to minimize $\mathcal{F}$ through optimal decision a, but it only knows the estimated request information $\hat{x}$.

To address this problem, we design a decision neural network to estimate the probability of a decision, as shown in Fig. 3. The decision network consists of an input layer, a hidden layer, and an output layer. We set the estimated vector $\hat{x}$ as the input of the decision network whose size is $(1, 3M)$. The input variables $\hat{x} = \{\hat{\mathbf{s}}, \hat{\mathbf{c}}, \hat{\mathbf{d}}\}$ are normalized according to variable type respectively through the input layer. The output layer consists of three paralleling activations to output the probability of resource allocation by the edge cloud which ensures that the result does not exceed the constraint due to the limited resource. Thus, the output vector with the size of $(1, 3M)$ can be given as

$$\pi(a|\hat{x}) = \{\pi_\beta, \pi_f, \pi_{\mathbf{p}}\} \tag{32}$$

The final decision variable $a = \{\beta, \mathbf{f^e}, \mathbf{p}\}$ can be calculated by the output of the decision network, which is given by

$$\Gamma(\pi) : \begin{cases} \beta = \pi_\beta \\ \mathbf{f^e} = \pi_f F_e \\ \mathbf{p} = \pi_p (p^{\max} - p^{\min}) + p^{\min} \end{cases} \tag{33}$$

However, there are no true values to supervise the learning of the neural networks in combinatorial optimization problems. Therefore, the target function $\mathcal{F}$ is used to be the loss function to supervise the parameter updating. Besides, in order to adapt to the worst-case situations with deviation δ, the loss consider the worst-case δ which can be given by the maximum

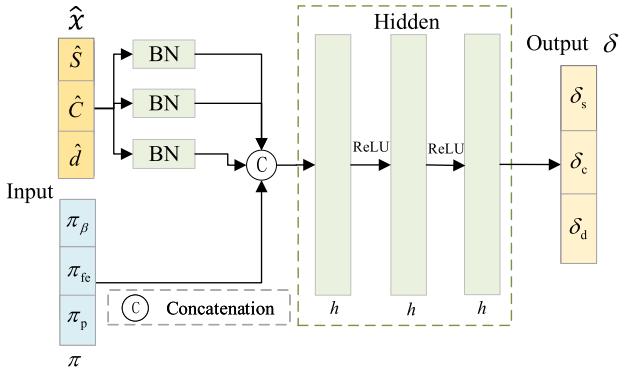


Fig. 4. Deviation network for the uncertain variable.

optimizer. So the decision network uses the output of the deviation network to calculate the worst-case loss. Due to the large variance of latency and energy consumption of different requests, the network is difficult to converge. The uniform allocation strategy is simple and normal situations, so it can be a baseline to normalize the objective. Note that it is not the optimal value or ground truth. We denote the uniform decision as a_{baseline}, so we can normalize the target function by

$$J(\varphi) = \frac{f(\hat{x}+\delta, a)}{f(\hat{x}, a_{\text{baseline}})} \tag{34}$$

where δ is the output of the deviation network. Then, we can get the gradient of $J(\varphi)$ as

$$\nabla_{\varphi} J(\varphi) = \nabla_{\varphi} f(\hat{x} + \Theta_{\theta}(\hat{x}, \Phi_{\varphi}(\hat{x})), \Phi_{\varphi}(\hat{x})) \tag{35}$$

where Φ and φ represent the decision network and its parameters and Θ and θ represent the deviation network and its parameters. Moreover, the parameter updating is as follows:

$$\varphi_{i+1} = \varphi_i + \eta_1 \nabla_{\varphi} J(\varphi) \tag{36}$$

where η_1 is the learning rate of the decision network.

The pseudocode of the decision network for DT-assisted task offloading is given in Algorithm 1. There are two decision networks in the DTRTO consisting of $\hat{x}$-decision network and δ-decision network, so the training process is divided into two phases. In the first phase, we train the neural network to get the $\Phi_{\varphi_{\hat{x}}}$, which is to predict the pre-decision probability of task offloading. Then, using the $\Phi_{\varphi_{\hat{x}}}$ to train the decision network for the second time, we can get the $\Phi_{\varphi_{\delta}}$.

B. Deviation Network for the Uncertain Variable

The purpose of the deviation network is to predict the error between the estimated values and true values in the worst scenario, and it is a maximization problem, which is given by

$$\mathcal{P}_{\mathbf{3}}: \max_{\delta_s, \delta_c, \delta_d} \alpha_1 T_{avg} + \alpha_2 E_{avg} \tag{37}$$

$$\text{subject to } C1: |\delta| \leq \epsilon \tag{38}$$

We denote the optimization objective function as

$$\mathcal{G}(\delta) = \alpha_1 T_{avg} + \alpha_2 E_{avg} \tag{39}$$

As shown in Fig. 4, the deviation network consists of an input layer with normalization, an MLP hidden layer with the nodes of h, and an output layer. Because the objective is related to the DT estimated value and the action, we set the input of the deviation network as the uncertain variable and the predictive decision probability, which can be expressed as

$$Input: \{\hat{x}, \pi(a|\hat{x})\} \tag{40}$$

The input variables $\hat{x} = \{\hat{s}_t, \hat{c}_t, \hat{d}_t\}$ are normalized according to variable type respectively through the input layer. Since $\pi(a|\hat{x})$ is the output of decision network, it can be input directly. Then, we output the deviation $\{\delta_s, \delta_m, \delta_d\}$. The purpose of the deviation network is to predict the uncertain deviation δ to maximize the objective $\mathcal{G}$. The problem is that we do not know the true deviation, so we can train the network by $\mathcal{G}$. However, the maximization problem is limited by the uncertain set, thus a regular term is added to limit the range of δ. The model is trained by adaptive moment estimation (Adam) optimizer. The loss function is penalized positively by the comparable magnitude of $\mathcal{G}$, making the deviation network develop in the direction of reducing δ. The formula of the loss

Algorithm 1 Decision Network for DT-Assisted Task Offloading

Input: Training set of DT estimated valuables D_x
Output: Network parameters that $\varphi_{\hat{x}}$ for $\hat{x}$-decision network $\Phi_{\varphi_{\hat{x}}}$ and φ_{δ} for δ-decision network $\Phi_{\varphi_{\delta}}$

Initialize network parameter φ_0 of decision network.
Set $\pi_{uniform}: \{\pi_{\beta,m} = \frac{1}{M}, \pi_{fe,m} = 0.5, \pi_{p,m} = 0.5, \forall m \in \mathcal{M}\}$
for episode in range(EPISODES) **do**
  for i in range(EPOCHS) **do**
    Random select $\hat{x} = D_{x,random}$
    Use decision network to output the probability of decision a by $\pi(a|\hat{x}) = \Phi_{\varphi}(\hat{x}) = \{\pi_{\beta}, \pi_{\mathbf{fe}}, \pi_{\mathbf{p}}\}$
    if episode = 0 **then**
      Set $\delta = \mathbf{0}$
    else
      Use deviation network to output $\delta = \Theta_{\theta}(\hat{x}, a_i)$
    end if
    Calculate the latency $T_{base} = T(\hat{x}, \pi_{uniform})$ and energy consumption $E_{base} = E(\hat{x}, \pi_{uniform})$ by uniform allocation decision $\pi_{uniform}$
    Normalize the latency and energy consumption by output allocation decision a
    $\begin{cases} T' = T(\hat{x}+\delta, \pi)/T_{base} \\ E' = E(\hat{x}+\delta, \pi)/E_{base} \end{cases}$
    Calculate loss function $J(\varphi_i) = \frac{1}{M}(\alpha_1 T' + \alpha_2 E')$
    Calculate the gradient of $\nabla_{\varphi_i} J(\varphi_i)$
    Update the parameter $\varphi_{i+1} = \varphi_i + \eta_1 \nabla_{\varphi_i} J(\varphi_i)$
  end for
  if episode = 0 **then**
    Save $\varphi_{\delta} = \varphi_{batchsize}$ for δ-decision network $\Phi_{\varphi_{\delta}}$
  end if
  Train the parameter θ for deviation network Θ_{θ}
end for
Save $\varphi_{\hat{x}} = \varphi_{batchsize}$ for $\hat{x}$-decision network $\Phi_{\varphi_{\hat{x}}}$

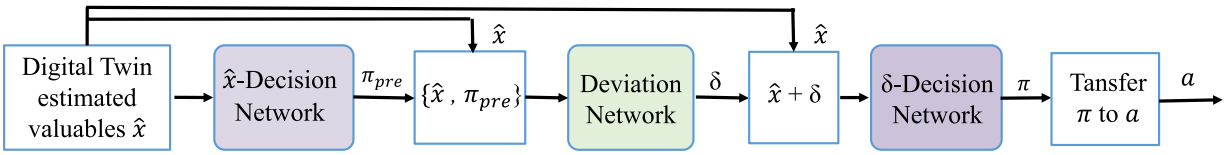


Fig. 5. Digital twin robust learning task offloading (DTRTO) scheme.

Algorithm 2 Deviation Network for Uncertain Variable

Input: Training set of DT estimated request D_x. Network parameter φ_δ of δ-Decision network Φ_{φ_δ}

Output: Network parameter θ for deviation network Θ_θ

for episode in range(EPISODES) **do**
  if episode=0 **then**
    Set parameter $\varphi = \varphi_\delta$ for decision network Φ_{φ_δ}
  else
    Set parameter $\varphi = \varphi_{\hat{x}}$ for decision network $\Phi_{\varphi_{\hat{x}}}$
  end if
  for i in range(EPOCHS) **do**
    Random select $\hat{x} = D_{x,random}$
    Use decision network to output the probability of decision a by $\pi(a|\hat{x}) = \Phi_\varphi(\hat{x}) = \{\pi_\beta, \pi_{\mathbf{fe}}, \pi_{\mathbf{p}}\}$
    Use deviation network to output $\delta = \Theta_\theta(\hat{x}, \pi)$
    Transfer π to a by $\Gamma(\pi)$
    Calculate average cost $\mathcal{G}(\delta) = -\frac{1}{M}(\alpha_1 T_{avg}(\hat{x}+\delta, a) + \alpha_2 E_{avg}(\hat{x}+\delta, a))$
    if $|\delta|_p > \epsilon$ **then**
      Calculate loss function $L(\theta_i) = \lambda(|\delta|_p - \epsilon)$
    else
      Calculate loss function $L(\theta_i) = -\mathcal{G}(\delta)$
    end if
    Calculate the gradient $\nabla_{\theta_i} L(\theta_i)$
    Update the parameter of deviation network $\theta_{i+1} = \theta_i + \eta_2 \nabla_{\theta_i} J(\theta_i)$
  end for
  Train the parameter φ for $\hat{x}$-Decision network $\Phi_{\varphi_{\hat{x}}}$
end for
Save $\theta_{\hat{x}} = \theta_{batchsize}$ for deviation network Θ_θ

function is described as follows:

$$L(\delta) = \begin{cases} \mathcal{G}(\delta), |\delta| \le \epsilon \\ \lambda(|\delta| - \epsilon), |\delta| > \epsilon \end{cases} \tag{41}$$

where $\lambda > 0$ is the additional penalty-weighted vector when the solution δ goes beyond ϵ.

$$\nabla_\theta J(\theta) = -\nabla_\theta \mathcal{G}(\Theta_\theta(\hat{x}, \Phi_\theta(\hat{x})), \Phi_\theta(\hat{x})) \tag{42}$$

Moreover, the parameter updating is as follows

$$\theta_{i+1} = \theta_i + \eta_2 \nabla_\theta J(\theta) \tag{43}$$

where η_2 is the learning rate of the deviation network.

The pseudocode of the deviation network for DT-assisted task offloading is given in Algorithm 2. Because the input of the deviation network is the estimated variables $\hat{x}$ and decision π, the training process of the deviation network should first use the decision network to predict the decision. For the first training phase, it uses the $\hat{x}$-decision network and for the other phase, it is trained using δ-decision network.

C. The DT-Assisted Robust Learning Task Offloading Scheme

The DTRTO scheme consists of the decision network and deviation network to solve the maximization problem and minimization problem respectively. According to network construction introduced before, the output decision a of the decision network is the input of the deviation network, while the output deviation δ is the key to calculating the loss $J(\varphi)$ of the decision network. Thus, they are trained and updated parameters alternately and support each other.

In the initial stage, the parameter φ_0 and θ_0 which are random initialized cause the network randomly direct the result forward. If such untrained parameters are used to forward the results and the results are used for another network, another network will be trained to error direction at the starting point. Therefore, to avoid disorientated training, we train the decision network whose loss is not considered the deviation δ, namely, setting $\delta = \mathbf{0}$ in the first episode. After a batchsize training, this decision pre-network could solve the minimization independently. Then, the decision pre-network is used to train the deviation network and they are trained alternately until to be converged.

Three networks are used to solve the optimization problem, as shown in Fig 5. The input is DT estimated variable $\hat{x}$ and the output is the decision a with the worst-case predicted deviation δ. Specifically, the decision network outputs the pre-decision $a_{pre} = \Phi_{\varphi_{\hat{x}}}(\hat{x})$ by input DT estimated $\hat{x}$. Then, the deviation network outputs $\delta = \Theta_\theta(\hat{x}, a_{pre})$ the deviation by pre-decision a_{pre} and DT estimated $\hat{x}$. We can get a decision a_{pre} at the first step, but this decision does not consider δ adequately. Thus, we input $\hat{x}+\delta$ into the pre-decision called δ-decision network which is trained at the first training phase. Note that the parameters of δ-decision network are different from $\hat{x}$-decision network. Finally, we can obtain the final decision $a = \Phi_{\varphi_\delta}(\hat{x}+\delta)$. The DTRTO is shown in Algorithm 3.

All the networks have the same construction except for the number of hidden nodes. Both input and output are $3M$. Given the number of hidden nodes h, the complexity of each network is $\mathcal{O}(12nh^2 log(6n+h+3nh)$. Therefore, the network has a lower computing complexity when predicting the decision. The main advantage of the proposed DTRTO is that it can infer the optimal task offloading resource allocation and power management scheme in DT when considering the uncertain deviation between DT and physical entities. Thus, the DTRTO improves the robustness of the DT-assisted mobile edge network.

Algorithm 3 Digital Twin-Assisted Robust Learning Task Offloading Scheme

Input: DT estimated variables $\hat{x}$

Output: Task offloading decision a

1: Using $\hat{x}$-decision network $\Phi_{\varphi_{\hat{x}}}$ to predict the pre-decision probability π_{pre} by estimated variables $\hat{x}$:$\pi_{pre} = \Phi_{\varphi_{\hat{x}}}(\hat{x})$.

2: Using deviation Network Θ_θ to predict the uncertain deviation by estimated variables and pre-probability: $\delta = \Theta_\theta(\hat{x}, \pi)$

3: Using decision Network to predict the final-decision probability by estimated variables and predicted deviation $\hat{x} + \delta$: $\pi = \Phi_{\varphi_\delta}(\hat{x} + \delta)$

4: Transfer the π to a: $a = \Gamma(\pi)$

5: Output task offloading decision a

V. Performance Evaluation

In this section, we evaluate the proposed DTRTO scheme. To perform large-scale performance analysis, we establish a parameter scheme based on real application scenarios and build a simulation system to verify the system performance.

A. Experimental Setting

We consider a DT-assisted edge network topology with one edge cloud and M mobile devices which are randomly active in an area of $1000m \times 1000m$. The distance d is generated by a random movement model with different speeds ranging from 15m/s to 20m/s. For the offloading scenario, we assume that the user $M \in (10, 50)$ requests task with the datasize and CPU cycle/unit in the range of $[100, 1000]$ MB and $[0, 100]$ cycle/MB respectively. We randomly generate requests for M mobile devices with data size and CPU cycle/unit in the range of $[0, s_{\max}]$ and $[0, c_{\max}]$, respectively [28]. In order to verify the robustness with uncertain deviations, we add the deviations by error ratio to the dataset which are generated by a normal distribution with mean and variance in the range of $(0, 0.1)$ and $(0, 0.05)$ respectively.

For the computing parameters, we set the computer frequency of the mobile device to $f^{loc} = 500$ MHz [8], and the maximum computer resource of the edge cloud is in the range of $F_e = [15, 60]$ GHz. For the transmission parameters, we set the maximum uplink transmission power of mobile devices to $p^{\max} = 200$ dBm, the bandwidth is $B = 5$ Mbps, the noise is $N_0 = -174$ dBm/Hz, and $PL(d_m) = -35.3 - 37.6 \log_{10} d_m$ [29]. The URLLC decoding error probability is set to $\epsilon_u = 10^{-7}$. The object effect vector is $\alpha_1 = 0.5$ and $\alpha_2 = 0.5$. A neural network consists of linear layers and an activation function with 128 hidden nodes. The learning rate is $\{10^{-2}, 10^{-3}, 10^{-4}, 5 \times 10^{-5}\}$. We set the maximum range of deviation $\epsilon = (0.01, 0.1)$ and λ is adjusted by ϵ.

To evaluate the performance of the scheme, we use the final optimization indicators which consists of average latency, average energy consumption, and average cost of mobile devices. Because uncertain deviations arose between the DT and physical entities, we employed a randomize deviation ratio in the test set to verify the robustness of the scheme.

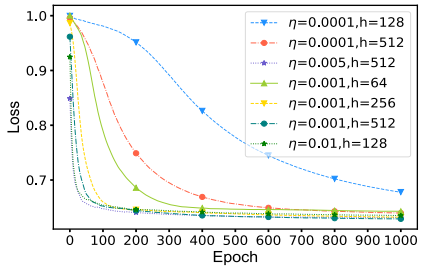


Fig. 6. The learning curve of δ-decision network.

B. Comparison Algorithms

Four benchmark algorithms are used to evaluate the proposed DTRTO algorithm: 1) **Random:** The task offloading and resource allocation strategy are generated randomly from $[0, 1]$. 2) **Uniform:** The uniform task offloading and resource allocation strategy that is defined as $\pi_{uniform}$: $\{\pi_{\beta,m} = \frac{1}{M}, \pi_{f_e,m} = 0.5, \pi_{p,m} = 0.5, \forall m \in \mathcal{M}\}$ 3) **LRCO:** A robust learning algorithm that only uses a minimizer network to predict decisions in the prediction phase [30]. 4) **Fixed deviation:** Pre-estimate a deviation between the DT and physical entities, it has the same structure as the decision network but fixes the estimated deviation [6].

C. Experimental Performance

1) DTRTO Scheme Convergence Analysis: The number of hidden nodes and learning rate are the primary coefficients of the decision network and the deviation network. We test the convergence with different coefficient settings in order to find the optimal configuration. Fig. 6 shows the learning curve of the δ-decision network which is trained at the first stage. It indicates the loss in each epoch with various learning rates η and hidden size h. Note that the loss decreases with fluctuation, so we smooth the curve by moving an average of window length 5 in order to observe the obvious trends. We can see that the network has converged after 1000 epochs except $\eta = 0.0001$ and higher learning rates accelerate the convergence speed. For the fixed learning rate, as the hidden sizes increase, the convergence speed up and the convergence value decreases. Thus, in the experiments, we chose $\eta = 0.0001, h = 512$ as the decision network's parameter to obtain the fastest convergence and the best result.

Fig. 7 shows the learning curve of the deviation network, which indicates the loss in each epoch with different learning rates. For the regular terms' deviation loss function, large differences occurred in the loss curves. It can be seen that the curve of 5×10^{-5}, 10^{-4}, 10^{-3} converge after 500 epochs, but the curve of 10^{-2} does not converge because the regular term is difficult to limit the development in the case of high learning rate. Thus, in the experiments, we set $\eta_2 = 0.0001$ as the optimal learning rate.

Fig. 8 shows the learning curve of the $\hat{x}$-decision network trained at the third stage. It indicates the two methods to train the network, relearning and based. Relearning means that the network randomly initializes the parameters and relearning. Based means that the network updates the parameter based on the δ-decision network. We can find that the based training

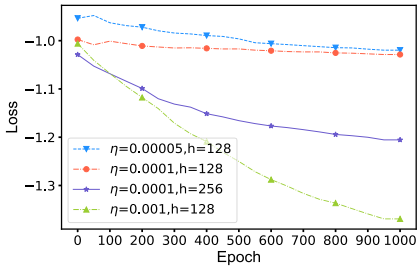


Fig. 7. The learning curve of deviation network.

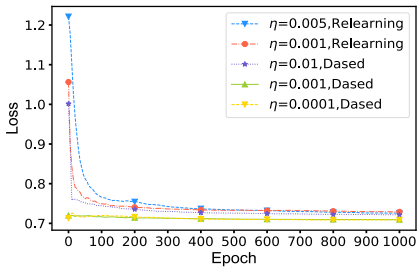


Fig. 8. The learning curve of $\hat{x}$-decision network.

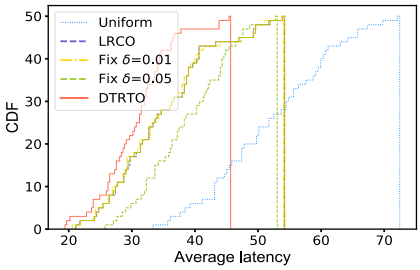


Fig. 9. The cumulative distribution function of task offloading average latency under five schemes.

network shows lower convergence values and that the learning curve mildly decreased. Moreover, it can be seen that the curve with $\eta = 0.0001$ is lower than $\eta = 0.001$. Thus, a lower learning rate is associated with optimal performance. Because the $\hat{x}$-decision is trained based on the δ-decision network, they are set to the same construct and hidden size of 512.

2) The CDF of Five Algorithms on the Average Latency, Average Energy Consumption, and Average Cost: Fig. 9-11 shows the cumulative distribution function (CDF) distribution of latency, energy consumption, and cost under different algorithms. It can be clearly seen that the performances of DTRTO whether measured by latency, energy consumption or cost are significantly better than that of other algorithms. Specifically, when comparing DTRTO to the fixed deviation scheme, DTRTO can better adapt to the situation with uncertain deviation. Furthermore, DTRTO is distributed in the lower region, meaning that DTRTO can obtain superior performance in the majority of cases.

3) The Impact of Different Parameters on Algorithms Performance: To verify the generalization of DTRTO, we simulate a variety of scenarios of task offloading consisting of different maximum datasize and required CPU cycles of mobile devices. Fig. 12-14 show the impact of maximum task data size s^{max}, the maximum CPU cycle/unit c^{max} and the

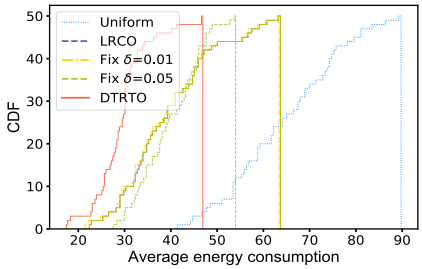


Fig. 10. The cumulative distribution function of task offloading average energy consumption under five schemes.

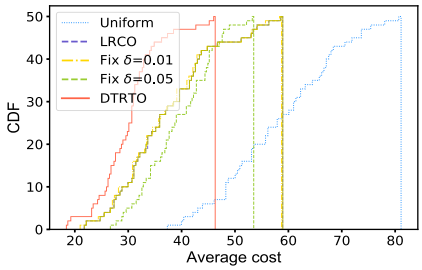


Fig. 11. The cumulative distribution function of task offloading average cost under five schemes.

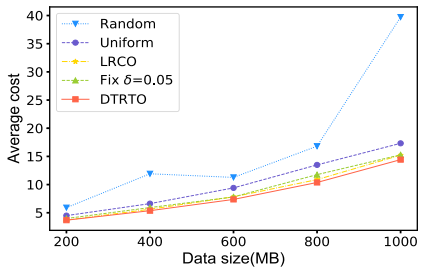


Fig. 12. The impact of maximum task data size for unit data on system average cost.

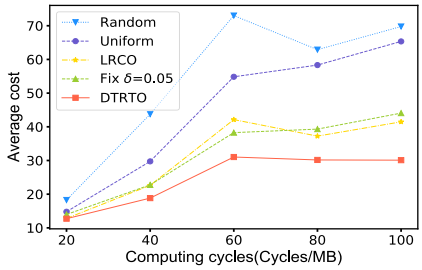


Fig. 13. The impact of maximum CPU cycle for unit data on system average cost.

maximum computing resources of edge cloud F_e on system average cost. Fig. 12 shows that the cost of DTRTO is the lowest. As the maximum task datasize increases, the average system cost increases because a larger data size requires more resources to offload. As shown in Fig. 13, the cost increases as the maximum unit CPU cycle increases and the DTRTO tends to be stable after 60 cycles compared to other schemes. The reason is that large CPU cycle requires a long computing time so the latency becomes the main factor of the task offloading scheme. Thus, DTRTO has a better stability when there are lots of complex tasks compared to other schemes. When changing the maximum computing frequency of the edge cloud, the

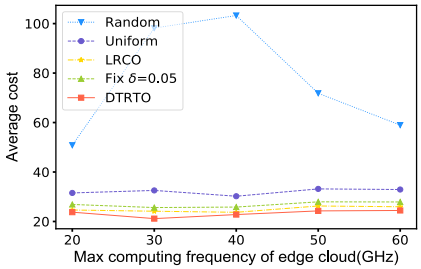


Fig. 14. The impact of maximum computing frequency of edge cloud on system average cost.

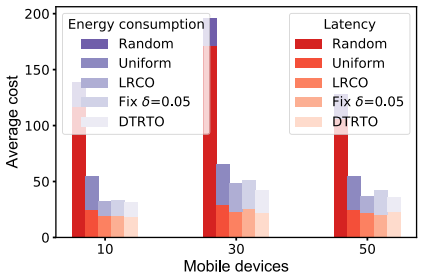


Fig. 15. The impact of mobile device number on the system average cost.

average cost shows a slight fluctuation in Fig. 14. The reason is that the maximum frequency influences the computing latency, but the transmission latency and energy consumption influence the cost in terms of a higher weight.

4) The Impact of the Number of Mobile Devices on Algorithms Performance: Since the input size of the network is $(1, 3M)$, the size of input sequences is decided by the number of mobile devices. So we construct some experiments in the case of various input sequences, as shown in Fig. 15, we assess the average energy, latency, and cost of five schemes. We find that in the case of different numbers of mobile devices, DTRTO shows lower average latency, energy consumption, and cost. Because we use different datasets of varied input sizes, the performance of the three groups reflect differences.

In summary, compared to other schemes, the DTRTO can reduce the latency and energy consumption in various scenarios when there are uncertain deviations. Thus, the DTRTO can improve the robustness of decision-making and address the stochastic asynchronization between the DT and physical entities.

VI. Conclusion

In this paper, we proposed a DT-assisted URLLC-enabled mobile edge network in 6G environment. To minimize the system latency and energy consumption, we formulated a task offloading problem considering the uncertain deviation between the DT and the physical entity. Moreover, we designed a DTRTO algorithm that includes decision networks and deviation network. We carried out extensive simulations to calculate the latency and energy consumption in various cases to verify the optimization performance of the system with deviation. The experimental results demonstrated that with system deviation, the DTRTO scheme can provide an effective offloading strategy with low latency and energy consumption.

References

[1] J. Zheng and M. Woźniak, "Design of quick search method for key feature images in mobile networks," *Mobile Netw. Appl.*, vol. 27, pp. 2524–2533, Dec. 2022.

[2] R. W. Liu et al., "STMGCN: Mobile edge computing-empowered vessel trajectory prediction using spatio-temporal multigraph convolutional network," *IEEE Trans. Ind. Informat.*, vol. 18, no. 11, pp. 7977–7987, Nov. 2022.

[3] M. Kumar et al., "BBNSF: Blockchain-based novel secure framework using RP2-RSA and ASR-ANN technique for IoT enabled healthcare systems," *Sensors*, vol. 22, no. 23, p. 9448, Dec. 2022.

[4] M. Woźniak, M. Wieczorek, and J. Siłka, "BiLSTM deep neural network model for imbalanced medical data of IoT systems," *Future Gener. Comput. Syst.*, vol. 141, pp. 489–499, Apr. 2023.

[5] P. Popovski et al., "Wireless access in ultra-reliable low-latency communication (URLLC)," *IEEE Trans. Commun.*, vol. 67, no. 8, pp. 5783–5801, Aug. 2019.

[6] D. Van Huynh, S. R. Khosravirad, A. Masaracchia, O. A. Dobre, and T. Q. Duong, "Edge intelligence-based ultra-reliable and low-latency communications for digital twin-enabled metaverse," *IEEE Wireless Commun. Lett.*, vol. 11, no. 8, pp. 1733–1737, Aug. 2022.

[7] W. Sun, H. Zhang, R. Wang, and Y. Zhang, "Reducing offloading latency for digital twin edge networks in 6G," *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, pp. 12240–12251, Oct. 2020.

[8] T. Do-Duy, D. Van Huynh, O. A. Dobre, B. Canberk, and T. Q. Duong, "Digital twin-aided intelligent offloading with edge selection in mobile edge computing," *IEEE Wireless Commun. Lett.*, vol. 11, no. 4, pp. 806–810, Apr. 2022.

[9] R. O. Ogundokun et al., "Non-orthogonal multiple access enabled mobile edge computing in 6G communications: A systematic literature review," *Sustainability*, vol. 15, no. 9, p. 7315, Apr. 2023.

[10] K. Wang et al., "Task offloading with multi-tier computing resources in next generation wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 2, pp. 306–319, Feb. 2023.

[11] Y. Dursun, F. Fang, and Z. Ding, "Hybrid NOMA based MIMO offloading for mobile edge computing in 6G networks," *China Commun.*, vol. 19, no. 10, pp. 12–20, Oct. 2022.

[12] J. He, S. Guo, M. Li, and Y. Zhu, "AceFL: Federated learning accelerating in 6G-enabled mobile edge computing networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 10, no. 3, pp. 1364–1375, May 2023.

[13] J. Yan, S. Bi, L. Duan, and Y.-J. A. Zhang, "Pricing-driven service caching and task offloading in mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 20, no. 7, pp. 4495–4512, Jul. 2021.

[14] T. Zhang and W. Chen, "Computation offloading in heterogeneous mobile edge computing with energy harvesting," *IEEE Trans. Green Commun. Netw.*, vol. 5, no. 1, pp. 552–565, Mar. 2021.

[15] Y. Sahni, J. Cao, L. Yang, and Y. Ji, "Multi-hop multi-task partial computation offloading in collaborative edge computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 5, pp. 1133–1145, May 2021.

[16] L. Li, T. Q. S. Quek, J. Ren, H. H. Yang, Z. Chen, and Y. Zhang, "An incentive-aware job offloading control framework for multi-access edge computing," *IEEE Trans. Mobile Comput.*, vol. 20, no. 1, pp. 63–75, Jan. 2021.

[17] X. Zheng, J. Lu, and D. Kiritsis, "The emergence of cognitive digital twin: Vision, challenges and opportunities," *Int. J. Prod. Res.*, vol. 60, no. 24, pp. 7610–7632, Dec. 2022.

[18] P. Almasan et al., "Network digital twin: Context, enabling technologies, and opportunities," *IEEE Commun. Mag.*, vol. 60, no. 11, pp. 22–27, Nov. 2022.

[19] J. Jagannath, K. Ramezanpour, and A. Jagannath, "Digital twin virtualization with machine learning for IoT and beyond 5G networks: Research directions for security and optimal control," in *Proc. ACM Workshop Wireless Secur. Mach. Learn.*, May 2022, pp. 81–86.

[20] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018.

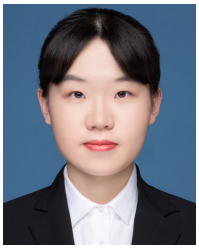
[21] T. H.-J. Uhlemann, C. Lehmann, and R. Steinhilper, "The digital twin: Realizing the cyber-physical production system for industry 4.0," *Proc. CIRP*, vol. 61, pp. 335–340, Jan. 2017.

[22] F. Tao, H. Zhang, A. Liu, and A. Y. C. Nee, "Digital twin in industry: State-of-the-art," *IEEE Trans. Ind. Informat.*, vol. 15, no. 4, pp. 2405–2415, Apr. 2019.

[23] W. Sun, N. Xu, L. Wang, H. Zhang, and Y. Zhang, "Dynamic digital twin and federated learning with incentives for air-ground networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 1, pp. 321–333, Jan. 2022.

[24] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Communication-efficient federated learning for digital twin edge networks in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 17, no. 8, pp. 5709–5718, Aug. 2021.
[25] B. Li, Y. Liu, L. Tan, H. Pan, and Y. Zhang, "Digital twin assisted task offloading for aerial edge computing and networks," *IEEE Trans. Veh. Technol.*, vol. 71, no. 10, pp. 10863–10877, Oct. 2022.
[26] G. Durisi, T. Koch, and P. Popovski, "Toward massive, ultrareliable, and low-latency wireless communication with short packets," *Proc. IEEE*, vol. 104, no. 9, pp. 1711–1726, Sep. 2016.
[27] Y. Polyanskiy, H. V. Poor, and S. Verdú, "Channel coding rate in the finite blocklength regime," *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2307–2359, May 2010.
[28] X. Xiong, K. Zheng, L. Lei, and L. Hou, "Resource allocation based on deep reinforcement learning in IoT edge computing," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1133–1146, Jun. 2020.
[29] D. Van Huynh, V.-D. Nguyen, S. Chatzinotas, S. R. Khosravirad, H. V. Poor, and T. Q. Duong, "Joint communication and computation offloading for ultra-reliable and low-latency with multi-tier computing," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 2, pp. 521–537, Feb. 2023.
[30] Z. Shao, J. Yang, C. Shen, and S. Ren, "Learning for robust combinatorial optimization: Algorithm and application," in *Proc. IEEE Conf. Comput. Commun. (IEEE INFOCOM)*, May 2022, pp. 930–939.

Yixue Hao (Member, IEEE) received the Ph.D. degree in computer science from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2017. He is currently an Associate Professor with the School of Computer Science and Technology, HUST. His current research interests include 5G networks, the Internet of Things, edge computing, edge caching, and cognitive computing.



Jiaxi Wang received the bachelor's degree in automation from the College of Control Science and Engineering, Shandong University, China, in 2021. She is currently pursuing the Ph.D. degree with the Embedded and Pervasive Computing (EPIC) Laboratory, School of Computer Science and Technology, Huazhong University of Science and Technology (HUST), China. Her research interests include edge computing and digital twin.

Dongkun Huo received the B.S. degree in computer science and technology from the School of Computer and Information Engineering, Henan University, China, in 2020, and the M.S. degree in computer technology from the School of Computer and Information Technology, Beijing Jiaotong University, China, in 2022. He is currently pursuing the Ph.D. degree with the Embedded and Pervasive Computing (EPIC) Laboratory, School of Computer Science and Technology, Huazhong University of Science and Technology (HUST), China. His research interests include MANET, edge computing, and digital twin.

Nadra Guizani received the Ph.D. degree from Purdue University in 2020. She is currently an Assistant Professor with the Department of Computer Science and Engineering, The University of Texas at Arlington. Her research interests include data analytics, artificial intelligence, blockchain, cybersecurity, network function virtualization, and machine learning in engineering education. She is an active member of ACM and the Computing Research Association.

Long Hu was a Visiting Student with the Department of Electrical and Computer Engineering, The University of British Columbia, from August 2015 to April 2017. He is currently an Associate Professor with the School of Computer Science and Technology, Huazhong University of Science and Technology (HUST), China. His research interests include the Internet of Things, software-defined networking, caching, 5G, body area networks, body sensor networks, and mobile cloud computing.

Min Chen (Fellow, IEEE) is currently a Full Professor with the School of Computer Science and Engineering, South China University of Technology. He is also the Director of the Embedded and Pervasive Computing (EPIC) Laboratory, Huazhong University of Science and Technology (HUST). Before joined HUST, he was an Assistant Professor with the School of Computer Science and Engineering, Seoul National University. His Google Scholar Citations reached more than 41,826 with an H-index of 96. His top paper was cited more than 4,361 times. He is a fellow of IET. He was a recipient of the IEEE Communications Society Fred W. Ellersick Prize in 2017, the IEEE Jack Neubauer Memorial Award in 2019, and the IEEE ComSoc APB Oustanding Paper Award in 2022. He is the founding Chair of the IEEE Computer Society Special Technical Communities on Big Data and the Chair of IEEE Globecom 2022 eHealth Symposium. He was selected as a Highly Cited Researcher from 2018 to 2022.