Joint Optimization of Model Retraining and Inference Services in DT-Assisted Edge Computing

Xuan Ai, Weifa Liang, *Senior Member, IEEE*, and Caiyi Liu

***Abstract*—With the advance of emerging digital twin (DT) technology, the combination of DT with edge intelligence brings great potential for high-fidelity, delay-sensitive inference services at the network edge. Due to the training data drift over time, the accuracy of an inference model degrades dramatically. To maintain and/or enhance its accuracy, the service model requires to be continuously retrained using newly generated update data. However, model retraining and inference services compete with each other for the limited computing resource in a mobile edge computing network (MEC), which may decrease user satisfaction with services due to unbearable service delays caused by insufficient resource supplies. Therefore, to ensure high fidelity of service models by choosing models for retraining while maximizing user satisfaction, it becomes a great challenge to allocate the limited computing resource in an MEC to both model retraining and inference services. In this paper, we investigate fidelity-aware, delay-sensitive services in a DT-assisted MEC network over a given time horizon. We study a novel user satisfaction maximization problem with the aim to maximize the long-term user satisfaction on services. We first formulate an integer linear programming (ILP) solution to its offline version. We then devise an online algorithm for the problem with a bounded expected cumulative regret, by leveraging an efficient prediction mechanism and a multi-armed bandit (MAB) based resource allocation strategy. Finally, we evaluate the performance of the proposed algorithm via simulations. Simulation results demonstrate that the proposed algorithm is promising, outperforming the comparison benchmarks.**

***Index Terms*—Digital twin, DT-assisted mobile edge computing, fidelity-aware and delay-sensitive inference services, joint resource allocation between model retraining and inference services, multi-armed bandit optimization, online algorithm, expected long-term regret.**

I. Introduction

DIGITAL twin (DT)-empowered inference services not only sustain high fidelity but also enable adaptive learning without costly real-world data collection. To achieve seamless integration between the digital and physical spaces, DTs are required to remain highly responsive to the dynamic changes of their physical counterparts by continuous synchronizations throughout their life cycles. Combining with technologies such as machine learning and artificial intelligence, DT has great potential to accelerate the advancement of edge intelligence [20]. The DT-assisted service model is anticipated to provide high-fidelity services based on the information extracted from a group of DTs [1]. However, DT synchronization is usually data-intensive and time-sensitive, and large amounts of resources are indispensable for the construction and maintenance of DT-assisted models in remote clouds [8]. Mobile edge computing (MEC) is powerful enough to support the updates of DTs by distributing communication and computing resources to the edge of core networks [14]. Furthermore, MEC is pivotal in boosting intelligent services empowered by DTs while meeting the stringent accuracy and delay requirements of users.

In this paper, we deal with fidelity-aware, delay-sensitive inference services in a DT-assisted MEC network, where each service model has multiple attributes, and each attribute is continuously fed by the update data from a DT, with DTs providing features derived from their corresponding physical objects to reflect the real-time states of the objects [5]. One illustrative example is a service model for road traffic monitoring that is built upon DTs of nearby mobile vehicles and roadside objects. The update data uploaded by a physical object will be transferred to its DT in the local edge computing network. The DT deployed in a cloudlet merges the update data with its existing data, and the processed update data is then transferred to the hosts of its models on demand for model retraining. As road traffic conditions always dynamically change over time, it will reduce the service accuracy of the road traffic monitoring if the service model is not retrained over time. Continuous DT synchronizations between DTs and their vehicles, and the model retraining by the updated DT data are necessary for accurate monitoring. The Age of Information (AoI) is a widely adopted metric to measure the freshness of DT data [25]. The service accuracy of a model built upon multiple DTs thus can be represented by the AoIs of its source DTs. Hence, it is critical to retrain a model continuously, using its updated source DT data in a rapidly changing environment. Fig. 1 is an illustration of fidelity-aware delay-sensitive inference services in a DT-assisted MEC network.

There are few studies of how continuous model retraining impacts inference services in MEC networks. For many real-world applications, although service models may be temporarily out of service during their retraining or maintenance, they can provide high-fidelity services after the retraining. To facilitate accurate inference services, in this paper, we consider a scenario where multiple service models are deployed in an MEC network, with model retraining and inference services

Received 13 June 2025; revised 29 September 2025; accepted 10 November 2025; approved by IEEE TRANSACTIONS ON NETWORKING Editor J.-W. Lee. Date of publication 18 November 2025; date of current version 8 January 2026. The work of Weifa Liang was supported by the Research Grants Council of the Hong Kong Special Administrative Region, China, under Grant CityU 11202723, Grant CityU 11202824, Grant C1042-23GF, Grant 7005845, Grant 8730094, and Grant 9380137. *(Corresponding author: Weifa Liang.)*

The authors are with the Department of Computer Science, City University of Hong Kong, Hong Kong, China (e-mail: xuanai2-c@my.cityu.edu.hk; weifa.liang@cityu.edu.hk; caiyiliu2-c@my.cityu.edu.hk).

Digital Object Identifier 10.1109/TON.2025.3632228

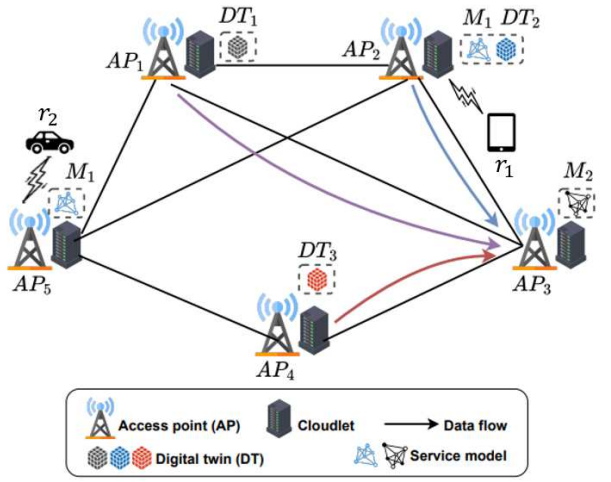


Fig. 1. An illustrative example of a DT-assisted MEC network with two service models M_1 and M_2. There are two instances of model M_1 deployed in cloudlet 2 and 5 that provide services to requests r_1 and r_2, while one instance of model M_2 deployed in cloudlet 3 is being retrained using its source DT update data that come from DT_1, DT_2, and DT_3, respectively.

simultaneously. However, model retraining competes for the limited computing resource with inference services, leading to a decrease in user satisfaction with services due to unbearable service delays caused by insufficient resource supplies.

To enhance service fidelity by choosing service models for retraining and to maximize user satisfaction by admitting as many user requests as possible, it poses the following challenges. First, as model retraining consumes computing resource, how to allocate the limited computing resource in an MEC to both model retraining and user request services judiciously? Second, although model retraining can bring more profits with improved service accuracy for future requests, it does compete with user service requests for limited computing resource, thereby reducing user request admissions. Then, which service models should be chosen for retraining without knowledge of future request arrivals? Finally, to maximize user satisfaction with services measured by both service accuracies and delays, how to accurately predict future requests and their requested service models, and how to assign the requests to cloudlets that host their requested models?

The novelty of the work lies in formulating a novel fidelity-aware, delay-sensitive inference service provisioning problem in a DT-assisted MEC network within a finite time horizon, with the aim to maximize the total user satisfaction with services. We design a novel MAB-based online algorithm for the problem, by leveraging the DT technology and an efficient prediction mechanism to predict dynamic resource allocation and future request types and resource demands. Our online algorithm is able to capture dynamic resource consumptions and explore non-trivial tradeoffs of resource allocations between model retraining and inference services. Unlike traditional MAB-based approaches that only rely on historical rewards, the proposed algorithm innovatively integrates a prediction mechanism supported by the DT technology, which is able to analyze historical request traces stored in the DT of each service model to predict request demands in the future.

The main contributions of this paper are presented as follows.

- We consider fidelity-aware delay-sensitive inference services in a DT-assisted MEC network, by exploring tradeoffs between model retraining and inference services, and formulate a novel user satisfaction maximization problem over a given time horizon, subject to resource capacities on the network. We also show NP-hardness of the problem and formulate an Integer Linear Programming (ILP) solution to its offline version.
- We devise an online algorithm for the user satisfaction maximization problem over a finite time horizon, by leveraging an efficient prediction mechanism and the multi-armed bandit optimization technique. The proposed algorithm is able to deliver a solution to the problem with a bounded expected cumulative regret.
- We evaluate the performance of the proposed online algorithm via simulations. Simulation results indicate that the proposed algorithm is promising.

The remainder of this paper is organized as follows. Section II surveys the related work on this topic. Section III introduces the system model. Section IV provides the problem formulation. Section V develops an online algorithm for the problem, and Section VI evaluates the performance of the proposed algorithm by simulations. Section VII concludes the paper.

II. Related Work

DT has emerged as a promising technology for high-fidelity inference service provisioning in edge computing, which has drawn lots of attention across diverse fields, including smart manufacturing, smart healthcare, autonomous driving, AR/VR, and so on. By incorporating DT technology into edge computing, fidelity-aware, delay-sensitive inference services can be seamlessly provided at the edge of core networks. For example, Wang et al. [27] studied a network-wide data collection scheme, which enables the information received by DTs to reflect the latest and consistent network-wide status with shorter and stable probing latency. Zhang et al. [32] proposed a mobility-aware service provisioning framework in a DT-assisted network, and devised an online algorithm with the aim to minimize the total cost of resources consumed to achieve accurate services, by deploying DT replicas in the network. Yang et al. [30] studied DT deployments of humans at the network edge and formulated a two-timescale accuracy-aware online optimization problem of optimizing both communication and computing resources. Zhao et al. [34] established a comprehensive distributed task offloading architecture by leveraging DT to predict future information of the system in order to minimize the total cost of task processing.

The deviation between a DT and its represented physical object becomes larger and larger if they do not synchronize with each other over time. To measure the freshness of a DT or a service model whose training data are from DTs, a metric – the Age of Information (AoI) is often adopted [12], where the AoI of a DT or a service model is the duration since the last update. To provide accurate services by a service model, the service model must be kept in a fresh state through retraining, using the update data. For instance,

Zhang et al. [33] provided an AoI-aware service framework in a DT-assisted MEC network for Digital Twin Network (DTN) slicing requests, and devised efficient algorithms for the expected cost minimization problem to maintain the freshness of both service models and their source DT data. Li et al. [17] considered synchronizations between DTs and their physical objects to minimize the average DT staleness with a given budget.

There are several studies of model retraining or inference services to ensure accurate service provisioning in MEC networks. For example, Wen et al. [22], [28] studied both single-device and multi-device edge inferences, which jointly considers sensing, computation, and communication for low-latency and high-accuracy intelligent service provisions in a system of one base station with a co-located edge server. Ren et al. [24] designed an asynchronous microservice provisioning framework with online learning in edge cloud networks, which addresses dynamic user requests, coupled with the diversity in microservice containers. Li et al. [19] studied the impact of model parameter size, by exploring nontrivial tradeoffs between retraining efficiency and inference efficiency through model compressions under time and memory constraints. Cai et al. [4] introduced a resource allocation problem for joint model retraining and inference services on a resource-limited edge server through adopting different resource allocation configurations, with the aim to maximize the long-term inference accuracy. Kong et al. [10] designed an adaptive on-device lightweight model update strategy to handle various adverse environments in a video analytics system, and to strive for a balance between inference accuracy and retraining latency. To cope with data drift, Zeng et al. [31] studied continuous model retraining to maintain inference accuracy and proposed a scheduling algorithm to allocate the limited resource to model retraining and inference services. Li et al. [14] investigated DT-empowered model retraining via continual learning in an MEC network, aiming to maximize the accuracy of service models, subject to the computing capacity on each cloudlet. Chen et al. [5] studied a DT-empowered service model selection problem, where the inference of a given feature can be implemented by different models with differential levels of accuracy. Liu et al. [21] quantified the freshness of models based on the AoI metric, and considered ML model deployments and freshness-sensitive task assignments, where the quality of inference services heavily depends on the freshness of their service models. They jointly optimized the quality of experience of users and the total profit of the service provider by leveraging the randomized rounding technique. Zhuang et al. [35] explored MEC-empowered large vision model services in wireless networks. They focused on the joint optimization of model inference and task offloading for user requests by considering service utility and energy consumption.

A closely related work to this study is applying the multi-armed bandit (MAB) optimization technique [3], [29]. For instance, Beytur et al. [3] proposed an online algorithm for a hierarchical inference system that combines the MAB and Lyapunov optimization techniques. They proposed a loss structure framework of bandits with expert advice, by developing an online learning algorithm to determine whether to upload tasks from mobile devices to an edge server for processing. Xu et al. [29] investigated an energy minimization problem for federated learning in an MEC network with uncertainty of client availability. They proposed an online algorithm for client selection with different contexts, which delivers a bounded regret by leveraging the MAB technique.

Different from the aforementioned studies that ignored either the resource limitation for model retraining and service inferences simultaneously, or the assistance provided to MEC networks by the prediction mechanism of digital twins, our work addresses these critical gaps.

III. Preliminaries

In this section, we introduce the system model, notions, and notations.

A. System Model

Consider an MEC network that is represented by an undirected graph $G = (N, E)$, where N is a set of Access Points (APs) and E is a set of links connecting APs. Each AP is co-located with a cloudlet, and the communication delay between an AP and its co-located cloudlet is negligible, as they are interconnected through an optic fiber cable [23]. Denote by n_j an AP or its co-located cloudlet with $1 \leq j \leq |N|$ if no confusion arises. The computing resource capacity on each cloudlet $n_j \in N$ is C_j. For each link $e \in E$, let d_e be the transmission delay of transmitting a unit data along link e. Assume that the MEC network operates in a discrete-time manner within a given time horizon $\mathbb{T}$ that is further divided into $|\mathbb{T}|$ equal time slots.

B. DTs of Objects, Service Models, and Service Requests

There is a set V of objects, and each object $v_i \in V$ has a digital twin DT_i, which is deployed in a cloudlet $h(DT_i) \in N$. DT_i needs to synchronize with its object v_i quite often to maintain the freshness of its state. There is a set $\mathcal{M}$ of service models that provide inference services for users. Let cloudlet $h(M_m)$ be the host of model M_m, in which its instance is deployed. Assume that each model $M_m \in \mathcal{M}$ has L_m attributes, and let $A_m = \{attr_{m,1}, \ldots, attr_{m,L_m}\}$ be the attribute set of M_m. The update data of the lth attribute, $attr_{m,l}$, of model M_m comes from a DT, e.g., DT_i of object v_i. Each model needs to be retrained, using the update data from its attributes when it is scheduled for retraining, and such a retraining takes a relatively short time. Thus, we assume that the retraining of each model lasts no more than the duration of one time slot.

Let R_t be a set of users requesting inference services on service models in $\mathcal{M}$ and $R_{m,t} \subseteq R_t$ be the set of users requesting services from model M_m at time slot t. Assume that there is a digital twin $DT(M_m)$ of service requests for model $M_m \in \mathcal{M}$, in which historical request traces requesting inference services on M_m are stored, which will be used for predicting this type of requests in the future. Specifically, the service request $r_k \in R_t$ is represented by a tuple $\langle j_k, s_k(t), m_k, \theta_k, D_k \rangle$, where j_k is the AP index at which the user of r_k is located at time slot t, $s_k(t)$ is the data size of the request input, m_k is the index of its requested service model, θ_k is the minimum service accuracy requirement of r_k with $0 < \theta_k \leq 1$, and D_k is the delay requirement of r_k. The user of request r_k is fully satisfied with

the service if the actual service delay is no greater than D_k; otherwise, the user is still partially satisfied with the service when the actual service delay is no greater than $\beta \cdot D_k$, where $\beta > 1$ is a constant. The user satisfaction with a service will be defined in Eq. (14) later. A service request is admitted only if all its requirements are met.

IV. Problem Formulation

In this section, we model inference service provisioning in a DT-assisted MEC network, formulate the problem precisely, and finally provide an ILP solution to the offline version of the problem.

A. DT Synchronization

As the freshness of an inference service is determined by its service model, the freshness of the service model is determined by its source DT data, and the freshness of DT data is ultimately determined by the synchronization frequencies between each DT and its object. For the DT, DT_i, of object $v_i \in V$, we assume that object v_i generates and uploads its update data to DT_i for synchronization with a pre-setting frequency, for example, v_i uploads its update data to DT_i every τ_i time slots, where $\tau_i \geq 1$ is a positive integer [11]. Denote by $AoI(DT_i, t)$ the Age of Information (AoI) of the data in DT_i at time slot t. We adopt a standard discrete AoI definition as follows [9].

$$AoI(DT_i, t) = \begin{cases} 1 & \text{if } DT_i \text{ is synchronized at } t-1, \\ AoI(DT_i, t-1) + 1 & \text{otherwise.} \end{cases} \tag{1}$$

Initially, assume that the AoI of digital twin DT_i is 1 at time slot 1.

Let $vol(DT_i, t)$ be the volume of cumulative data stored in DT_i at time slot t, and we have

$$vol(DT_i, t) = \begin{cases} vol(DT_i, t-1) + vol(v_i, t) \\ \quad \text{if } DT_i \text{ synchronizes at time } t, \\ vol(DT_i, t-1) \qquad \text{otherwise.} \end{cases} \tag{2}$$

where $vol(v_i, t)$ is the volume of the update data uploaded by object v_i at time slot t, which are generated by v_i since its last uploading for synchronization. Furthermore, each DT DT_i adopts an initial synchronization at time slot 0.

B. Freshness and Accuracy of a Service Model

Consider a set $\mathcal{M}$ of models of inference services, of which service accuracy can be measured by the freshness of models. The freshness of a service model $M_m \in \mathcal{M}$ is determined by when it is retrained and the freshness of its training data (its attributes' update data). It is noticed that the freshness of a model will significantly degrade if the model has not been retrained for a prolonged period, and its service fidelity (accuracy) will decrease too. To maintain high inference accuracy of a service model, the model needs to be retrained often if there is any update on its attribute data since its last retraining. It can be seen that the freshness of the attribute data of model M_m is determined by the AoI of the update data in its source DTs, which is defined by Eq. (1), i.e., each DT keeps a time stamp, which is the time slot when the update data from its object is received and merged into the DT.

Since different attribute data of a model M_m towards its service accuracy is different, a positive weight $\omega_{m,l}$ is adopted to represent the contribution of attribute $attr_{m,l}$ to the service accuracy of M_m, and $\sum_{attr_{m,l} \in A_m} \omega_{m,l} = 1$, where A_m is the set of all attributes of model M_m [15], [18]. If the data of $attr_{m,l}$ in A_m is supplied by DT_i, the AoI of the data of $attr_{m,l}$ in model M_m at time slot t is defined as follows.

$$AoI(attr_{m,l}, t) = AoI(DT_i, t). \tag{3}$$

Model freshness: The freshness $F(M_m, t)$ of model M_m at time slot t is defined as follows.

$$F(M_m, t) = \begin{cases} \sum\limits_{attr_{m,l} \in A_m} \omega_{m,l} \cdot AoI(attr_{m,l}, t) \\ \quad \text{if } M_m \text{ is retrained at time } t, \\ F(M_m, t-1) + 1 \qquad \text{otherwise.} \end{cases} \tag{4}$$

The metric to measure the freshness of a model is the weighted sum of the AoIs of its attribute data after its retraining at time slot t; otherwise, $F(M_m, t)$ will increase by 1 without retraining. It can be seen that a large value of $F(M_m, t)$ indicates that model M_m is stale and needs retraining soon. Assume that each model $M_m \in \mathcal{M}$ takes an initial training at time slot 0.

Model accuracy: We assume that the service accuracy (or fidelity) of a model is proportional to its freshness. The service accuracy decay is proportional to the duration that the model has not been trained since its last retraining, and this decay further accelerates with the increase on the duration. Thus, the service accuracy of a model is represented by a submodular non-increasing function $g(\cdot)$ through mapping its freshness to its service accuracy.

Following the AoI definition of the attribute data in model M_m, the larger the freshness value $F(M_m, t)$ of the model, the lower its service accuracy at time slot t. Let $acc_m(t)$ be the service accuracy of model M_m at time slot t, which is a submodular function defined as follows.

$$acc_m(t) = g(F(M_m, t)/|\mathbb{T}|), \tag{5}$$

e.g., $g(x) = \frac{a - a^x}{a}$, where $a > 1$ is constant, and the value of x represents the duration that model M_m has not been trained since its last retraining with $0 \leq F(M_m, t) \leq |\mathbb{T}|$ and $0 \leq x \leq 1$. Thus, if model M_m is retrained at time slot t, then its gain $\Delta acc_m(t)$ of service accuracy at time slot t is defined as follows.

$$\begin{aligned} \Delta acc_m(t) = {} & g(F(M_m, Eq.\ (2)t)/|\mathbb{T}|) \\ & - g((F(M_m, t-1) + 1)/|\mathbb{T}|), \end{aligned} \tag{6}$$

due to the fact that the freshness of model M_m will be $F(M_m, t-1) + 1$ at time slot t without retraining.

C. Computing Resource for a Model Retraining

Given time slot t, for each model $M_m \in \mathcal{M}$, assume that $\hat{t}_m$ ($\hat{t}_m < t$) is the time slot of its last retraining. Let $A_m(t) \subseteq A_m$ be the attribute set of model M_m, in which the data have been updated since its last retraining at time slot $\hat{t}_m$. It can be seen that if model M_m is chosen for retraining at time slot t, then $A_m(t) \neq \emptyset$. It is also noticed that the data of

attribute $attr_{m,l} \in A_m \backslash A_m(t)$ has not been updated since the last retraining of model M_m, and there is no need to transmit the DT data of $attr_{m,l}$ to the host of model M_m. Due to incremental retraining, the amount $C_{m,t}^{train}$ of computing resource demanded for the retraining of model M_m at time slot t is proportional to the cumulative volume of the update data from its attribute sources since its last retraining. We have

$$C_{m,t}^{train} = \sum_{attr_{m,l} \in A_m(t)} \xi_m (vol(attr_{m,l}, t) - vol(attr_{m,l}, \hat{t}_m)), \tag{7}$$

where ξ_m is the amount of computing resource needed per unit data to ensure that the retraining of model M_m can be finished within a single time slot, and $vol(attr_{m,l}, t) = vol(DT_i, t)$ is the amount of the update data of attribute $attr_{m,l}$ that is fed by DT_i, which is similar to the definition in Eq. (3). Notice that $C_{m,t}^{train}$ is given as all information of previous time slots can be revealed by time slot t.

D. User Request Admissions and Model Retraining

Recall that R_t is the set of requests for inference services at time slot t and $R_{m,t}$ is the set of requests for service model M_m, then $R_t = \cup_{M_m \in \mathcal{M}} R_{m,t}$. Given time slot t, if a model M_m is retrained at time slot t, then all requests in $R_{m,t}$ will not be admitted due to the unavailability of model M_m at time slot t. Let $y_{m,j,t}$ be a binary variable indicating whether model M_m is retrained in cloudlet n_j at time slot t ($y_{m,j,t} = 1$) or not ($y_{m,j,t} = 0$). Let $x_{k,j,t}$ be a binary variable indicating whether request $r_k \in R_t$ is assigned to a model instance M_{m_k} hosted by cloudlet n_j for processing at time slot t ($x_{k,j,t} = 1$) or not ($x_{k,j,t} = 0$). We have

$$\sum_{j=1}^{|N|} x_{k,j,t} + \sum_{j=1}^{|N|} y_{m_k,j,t} \leq 1, \ \forall r_k \in R_t, \ \forall \ t \in \mathbb{T} \tag{8}$$

which ensures that request r_k is either admitted or rejected at time slot t. If admitted, its requested model M_{m_k} should not be retrained at time slot t.

If request r_k is assigned to cloudlet n_j, its service delay consists of the uploading delay, the transmission delay of input data, and the processing delay at its requested model host as follows.

Uploading delay: The data uploading rate $\zeta_{k,t}$ of the user of request $r_k \in R_t$ to AP n_{j_k} can be calculated as follows.

$$\zeta_{k,t} = B_{j_k} \cdot \log \left(1 + \frac{PX_k \cdot H_{k,j_k}}{\sigma^2}\right), \tag{9}$$

where B_{j_k} is the bandwidth capacity on AP n_{j_k}, PX_k is the transmission power of the device of r_k, H_{k,j_k} is the channel power gain by uploading data from the user of r_k, and σ is the noise power spectral density.

The uploading delay of request $r_k \in R_t$ at time slot t is

$$d\_upload(r_k, t) = \frac{s_k(t)}{\zeta_{k,t}}, \tag{10}$$

where $s_k(t)$ is the data volume of the input of request r_k.

Transmission delay: When the user of r_k issues the request via a gateway cloudlet n_{j_k}, the request and its input will be transferred from gateway n_{j_k} to a cloudlet n_j in which model M_m is deployed. Let $d\_trans(r_k, n_j, t)$ be the data transmission delay. Then

$$d\_trans(r_k, n_j, t) = s_k(t) \cdot \sum_{e \in P(n_{j_k}, n_j)} d_e, \tag{11}$$

where $P(n_{j_k}, n_j)$ is a shortest path in G between cloudlets n_{j_k} and n_j, and d_e is the transmission delay per unit data along link $e \in E$.

Processing delay: Let C_m^{inf} be the amount of computing resource demanded by inferences on service model M_m. The processing delay of user request r_k for model M_{m_k} in cloudlet n_j is

$$d\_proc(r_k, n_j, t) = \frac{s_k(t)}{C_{m_k}^{inf}}, \tag{12}$$

where $s_k(t)$ is the input data size of r_k.

The end-to-end delay experienced by the user of request $r_k \in R_t$ is

$$\begin{aligned} delay(r_k, n_j, t) = {} & d\_upload(r_k, t) + d\_trans(r_k, n_j, t) \\ & + d\_proc(r_k, n_j, t). \end{aligned} \tag{13}$$

Metric of user satisfaction on services: User satisfaction with a service is usually determined by both the service delay and the service accuracy, which is modeled as follows. Sometimes, though the delay requirement D_k of request $r_k \in R_t$ may not be met, the request user may still satisfy the service if the service delay is within its tolerable delay range [16]. Hence, user satisfaction with a service is inversely proportional to the experienced service delay, which is measured by the following function.

$$d(r_k, n_j, t) = \begin{cases} 1 - \dfrac{[delay(r_k, n_j, t) - D_k]^+}{\beta \cdot D_k} \\ \qquad \text{if } delay(r_k, n_j, t) \leq \beta \cdot D_k, \\ 0 \qquad \text{otherwise}, \end{cases} \tag{14}$$

where $[x]^+ = \max\{x, 0\}$, and $\beta > 1$ is a constant, measuring the user's tolerance of the delay violation. It can be seen that the maximum value $d(r_k, n_j, t)$ of user satisfaction with the service delay is 1, and 0 in the worst case. When $delay(r_k, n_j, t) \leq D_k$, $d(r_k, n_j, t) = 1$, indicating that the delay requirement of request r_k can be fully satisfied. When $D_k < delay(r_k, n_j, t) \leq \beta \cdot D_k$, $\frac{1}{\beta} \leq d(r_k, n_j, t) < 1$, which indicates that although the service delay is acceptable, the user satisfaction of r_k on the service will decrease. When $delay(r_k, n_j, t) > \beta \cdot D_k$, $d(r_k, n_j, t) = 0$, this implies that the user of r_k is not satisfied with the service totally.

For the service accuracy requirement θ_k of request $r_k \in R_t$, this requires that the service accuracy $acc_{m_k}(t)$ of its requested model M_{m_k} at time slot t is no less than its minimum accuracy requirement θ_k. In other words, if request r_k is admitted at time slot t, then the service accuracy $acc_{m_k}(t)$ ($= g\left(\frac{F(M_{m_k}, t)}{|\mathbb{T}|}\right)$) of its requested model M_{m_k} must meet $acc_{m_k}(t) \geq \theta_k$ with $0 \leq acc_{m_k}(t) \leq 1$. The satisfaction $U(r_k, n_j, t)$ of the user of request $r_k \in R_t$ with his requested service model M_{m_k} hosted by cloudlet n_j is a linear combination of the service delay and the service accuracy, which can be expressed as follows.

$$U(r_k, n_j, t) = \alpha \cdot acc_{m_k}(t) + (1 - \alpha) \cdot d(r_k, n_j, t), \tag{15}$$

where α is a balancing coefficient between service accuracy and service delay with $\alpha \in [0,1]$. A larger α implies that users are more sensitive to service accuracy; otherwise, users are more sensitive to service delays.

E. Resource Allocations to Model Retraining and Inference Services

As both model retraining and inference services take place in each cloudlet at the same time, the limited computing resource in the cloudlet needs to be allocated to these two different activities dynamically. Within the given time horizon, in some time slots, there are fewer request arrivals. In this case, it should allocate more resource for model retraining. However, in other time slots, some models have not been trained for a while, their service fidelity degrades over time, and they need to be retrained. A fractional resource allocation of each cloudlet to either of the two activities exhibits randomness, and this fraction cannot be fixed. To capture the fractional resource allocation randomness, assume that there is a finite set Ω of resource allocation ratios, in which each ratio $\delta \in \Omega$ corresponds to a resource allocation policy with $0 < \delta < 1$. That is, at each time slot, we randomly choose a ratio in Ω with a given probability to allocate a fractional resource of each cloudlet to model retraining and inference services, while the probability of picking a ratio is obtained by analyzing historical information - the cumulative benefit brought by the ratio in the past, where set Ω is defined as follows. For instance, resource allocation ratios in Ω are quantized to $|\Omega|$ levels by uniform quantization over the interval $[0, \delta_{\max}]$, and the step size between quantization levels is uniform and set as $\frac{\delta_{\max}}{|\Omega|}$ with $\delta_{\max}$ (≤ 1) is the maximum value of δ.

Given a ratio $\delta \in \Omega$ at time slot t, there is a corresponding resource allocation approach: the amounts of computing resource in each cloudlet n_j allocated to model retraining and inference services are $\delta \cdot C_j$ and $(1-\delta) \cdot C_j$, respectively. For each cloudlet $n_j \in N$, recall that $x_{k,j,t}$ and $y_{m,j,t}$ are binary variables, we have

$$\sum_{M_m \in \mathcal{M}} C_{m,t}^{train} \cdot y_{m,j,t} \leq \delta \cdot C_j, \ \forall t \in \mathbb{T} \tag{16}$$

$$\sum_{r_k \in R_t} C_{m_k}^{inf} \cdot x_{k,j,t} \leq (1-\delta) \cdot C_j, \ \forall t \in \mathbb{T} \tag{17}$$

$$\sum_{j=1}^{|N|} (x_{k,j,t} + y_{m_k,j,t}) \leq 1, \quad \forall r_k \in R_t, \ \forall t \in \mathbb{T}, \tag{18}$$

where In eq. (16) and In eq. (17) are the resource budgets on model retraining and inference services, respectively. Constraint (18) ensures that each model is either in training or inference services at time slot t, exclusively.

We formulate fidelity-aware, delay-sensitive inference services in a DT-assisted MEC network within a given time horizon $\mathbb{T}$ as the user satisfaction maximization problem, by exploring non-trivial tradeoffs of allocating limited computing resource to both model retraining and inference services simultaneously, with the aim to maximize the total user satisfaction, i.e.,

$$\text{Maximize} \quad \sum_{t=1}^{|\mathbb{T}|} \sum_{r_k \in R_t} \sum_{j=1}^{|N|} U(r_k, n_j, t) \cdot x_{k,j,t}, \tag{19}$$

where $U(r_k, n_j, t)$ is defined in Eq. (15).

F. Problem Definition

Definition 1: Given an MEC network $G = (N, E)$, a finite time horizon $\mathbb{T}$, a set Ω of potential resource allocation ratios, a set V of objects with deployed DT_i of each object $v_i \in V$ in a cloudlet $h(DT_i)$, a set $\mathcal{M}$ of models with an attribute set A_m for each model M_m, assume that the data of the lth attribute $attr_{m,l}$ of model M_m comes from the DT of an object. Let R_t be a set of requests requesting inference services on different models in $\mathcal{M}$ at time slot $t \in \mathbb{T}$, and each request $r_k \in R_t$ is represented by a tuple $\langle j_k, s_k(t), m_k, \theta_k, D_k \rangle$. The user satisfaction maximization problem in G is to maximize the total user satisfaction in formula (19) over time horizon $\mathbb{T}$, subject to computing resource capacities on cloudlets.

Theorem 1: The metric defined by Eq. (4) to measure the freshness of each model $M_m \in \mathcal{M}$ is feasible, assuming that the weight $\omega_{m,l}$ of the lth attribute of each model $M_m \in \mathcal{M}$ is given, where $0 \leq \omega_{m,l} \leq 1$, $1 \leq l \leq L_m$, and $\sum_{l=1}^{L_m} \omega_{m,l} = 1$.

According to the definition of freshness $F(M_m, t) \geq 0$ of model $M_m \in \mathcal{M}$ at each time slot $\forall\, t \in \mathbb{T}$ by Eq. (4), if model M_m is not chosen for retraining at time slot t, the data used for its last retraining becomes stale gradually, and the AoI of model M_m increases by 1, i.e., $F(M_m, t) = F(M_m, t-1)+1$. Otherwise, if model M_m is chosen for retraining at time slot t, its freshness $F(M_m, t)$ will be updated to $\sum_{attr_{m,l} \in A_m} \omega_{m,l} \cdot AoI(attr_{m,l}, t)$. The definition by Eq. (4) is feasible if and only if

$$\sum_{attr_{m,l} \in A_m} \omega_{m,l} \cdot AoI(attr_{m,l}, t) < F(M_m, t-1) + 1, \tag{20}$$

which ensures that a retrained model will be fresher and have a higher fidelity. To show that In eq. (20) always holds, we proceed as follows.

Denote by $A_m(t) \neq \emptyset$ the set of source attributes of model M_m that have been updated since the last retraining of M_m. Let $\hat{t}_m$ be the time slot of the last retraining of M_m with $\hat{t}_m < t$. We then have

$$\begin{aligned}
&F(M_m, t-1) + 1 - \sum_{attr_{m,l} \in A_m} \omega_{m,l} \cdot AoI(attr_{m,l}, t) \\
&= F(M_m, \hat{t}_m) + t - \hat{t}_m \\
&\quad - \sum_{attr_{m,l} \in A_m} \omega_{m,l} \cdot AoI(attr_{m,l}, t) && (21) \\
&= \sum_{attr_{m,l} \in A_m} \omega_{m,l} \cdot AoI(attr_{m,l}, \hat{t}_m) + t - \hat{t}_m \\
&\quad - \sum_{attr_{m,l} \in A_m} \omega_{m,l} \cdot AoI(attr_{m,l}, t) && (22) \\
&= \sum_{attr_{m,l} \in A_m} \omega_{m,l} \cdot (AoI(attr_{m,l}, \hat{t}_m) \\
&\quad - AoI(attr_{m,l}, t) + t - \hat{t}_m) && (23) \\
&= \sum_{attr_{m,l} \in A_m(t)} \omega_{m,l} \cdot (AoI(attr_{m,l}, \hat{t}_m) \\
&\quad - AoI(attr_{m,l}, t) + t - \hat{t}_m) && (24) \\
&< 0, && (25)
\end{aligned}$$

where Eq. (21) holds as $\hat{t}_m$ is the time slot of the last retraining and $F(M_m, t-1) - F(M_m, \hat{t}_m) = t-1-\hat{t}_m$, Eq. (22) holds as the freshness of M_m was updated by Eq. (4), Eq (23) holds as

TABLE I
TABLE OF NOTATIONS

Notation	Descriptions
$G = (N, E)$	An MEC network with a set N of APs and a set E of links
B_j, C_j	The bandwidth capacity of AP n_j, and the computing resource capacity on its co-located cloudlet n_j
d_e	The transmission delay of transmitting a unit data along link $e \in E$
$\mathbb{T} = \{1, \ldots, \|\mathbb{T}\|\}$	A finite time horizon
V	A set of objects
DT_i, $h(DT_i)$	The digital twin of object $v_i \in V$ and its hosting cloudlet
$\mathcal{M}$, $h(M_m)$	A set of service models and the cloudlet hosting model $M_m \in \mathcal{M}$
L_m, $attr_{m,l}$	The number of attributes of model M_m and its lth attribute
A_m, $A_m(t)$	The attribute set of model M_m, and the subset of attributes that has been updated since the last retraining of M_m until time slot t
R_t, $R_{m,t}$	The sets of users requesting services from models in $\mathcal{M}$ and from model M_m at time slot t
DT_{M_m}	The DT of service requests for model M_m
$\langle j_k, s_k(t), m_k, \theta_k, D_k \rangle$	The request of user $r_k \in R_t$, where j_k is its located AP index, $s_k(t)$ is its input data size, m_k is its requested model index, θ_k is its service accuracy requirement, and D_k is its delay requirement
β	A constant to measure the user tolerance of actual service delay violation
τ_i	A pre-setting synchronization frequency for DT_i
$AoI(DT_i, t)$	The AoI of the data in DT_i at time slot t
$vol(DT_i, t)$	The volume of cumulative data stored in DT_i at time slot t
$vol(v_i, t)$	The volume of the update data uploaded by object v_i at time slot t
$\omega_{m,l}$	A positive weight representing the contribution of attribute $attr_{m,l}$ to the accuracy of M_m
$AoI(attr_{m,l}, t)$	The AoI of the data of $attr_{m,l}$ in model M_m at time slot t
$vol(attr_{m,l}, t)$	The volume of cumulative data in attribute $attr_{m,l}$ at time slot t
$F(M_m, t)$	The freshness of model M_m at time slot t
$g(\cdot)$	A submodular non-increasing function mapping model freshness to service accuracy
$acc_m(t)$, $\Delta acc_m(t)$	The service accuracy of model M_m at time slot t, and its accuracy increment if retrained at time slot t
$C_{m,t}^{train}$	The amount of computing resource demanded for the retraining of model M_m at time slot t
ξ_m	The amount of computing resource needed by the retraining of model M_m per unit data
C_m^{inf}	The amount of computing resource demanded for inference service on model M_m
$x_{k,j,t}$	A binary variable indicating whether the request of user $r_k \in R_t$ is assigned to cloudlet n_j for processing
$y_{m,j,t}$	A binary variable indicating whether model M_m is retrained in cloudlet n_j at time slot t
$\zeta_{k,t}$	The uploading rate of user $r_k \in R_t$ to AP n_{j_k}
$d\_upload(r_k, t)$	The uploading delay of user $r_k \in R_t$ at time slot t
$d\_trans(r_k, n_j, t)$	The transmission delay of the request of user $r_k \in R_t$ from its gateway cloudlet n_{j_k} to cloudlet n_j
$d\_proc(r_k, n_j, t)$	The processing delay of the request of user $r_k \in R_t$ at cloudlet n_j
$delay(r_k, n_j, t)$	The end-to-end delay experienced by the request of user $r_k \in R_t$
$d(r_k, n_j, t)$	User satisfaction with delay if request of user $r_k \in R_t$ is processed on cloudlet n_j
α	A balancing coefficient between the service delay and service accuracy on user satisfaction
$U(r_k, n_j, t)$	The overall satisfaction of user $r_k \in R_t$ with its request processed by model M_{m_k} hosted by cloudlet n_j
δ, δ_t	A resource allocation ratio and the chosen ratio at time slot t
Ω	A set of potential resource allocation ratios
$p_{\delta,t}$	The probability of selecting resource allocation ratio δ at time slot t
$\mu_{\delta,t}$, $\tilde{\mu}_{\delta,t}$	The actual reward and estimated reward of selecting resource allocation ratio δ at time slot t
$\gamma_{m,t'}$	The percentage of admitted requests in $R_{m,t'}$ at previous time slot $t' \leq t$
S	The sliding window of future time slots
$\Delta U_{m,t}$	The potential profit of model M_m in the future time window S if being retrained at time slot t

$t - \hat{t}_m = \sum_{attr_{m,l} \in A_m(t)} \omega_{m,l} \cdot (t - \hat{t}_m)$, (24) holds as the AoI of an attribute will increase by $t - \hat{t}_m$ if it has not been updated since $\hat{t}_m$, and Ineq (25) holds as the AoI of each attribute in $A_m(t)$ will not increase by $t - \hat{t}_m$ due to the update. The theorem then follows.

Theorem 2: The user satisfaction maximization problem in an MEC network $G = (N, E)$ is NP-hard.

We show the NP-hardness of the user satisfaction maximization problem in an MEC network by a reduction from the classic NP-hard problem – the maximum-profit generalized assignment problem (GAP). Given a set B of bins with each bin b having capacity C_b, there is a set I of items. Assigning item i to bin $b \in B$ has a profit $p_{i,b}$ with size of $s_{i,b}$. The maximum profit GAP is to assign as many items in I as possible to $|B|$ bins such that the total profit is maximized, subject to the capacity on each bin. We consider a special case of the problem, where the time horizon $\mathbb{T}$ consists of one time slot only, no models are selected for retraining, and the service delay and accuracy requirements of all users are satisfied. Each cloudlet n_j corresponds to bin j with capacity C_j, and each request is treated as an item. When request $r_k \in R_t$ is admitted in cloudlet n_j, the associated item is assigned to bin j, which consumes the amount $C_{m_k}^{inf}$ of computing resource of bin j and brings a profit $U(r_k, n_j, t)$. We aim to find a request assignment to maximize the total profit of admitted requests, subject to computing capacity on each cloudlet. It can be seen that this special problem is equivalent to the maximum profit GAP. As the maximum profit GAP is NP-hard [6], the user satisfaction maximization problem is NP-hard, too.

For the sake of convenience, the notations used in this paper are summarized in Table I.

G. ILP Formulation of the Offline Version of the User Satisfaction Maximization Problem

Let R_t be the set of requests requesting services of models in $\mathcal{M}$ at time slot t. The binary variable $x_{k,j,t}$ indicates

whether request $r_k \in R_t$ is admitted and assigned to a cloudlet n_j that hosts model M_{m_k} at time slot t. Let $\mathcal{M}(t)$ be a subset of models of which there are updates on their attribute data since their last retraining, i.e., model $M_m \in \mathcal{M}(t)$ when $A_m(t) \neq \emptyset$. The binary variable $y_{m,j,t}$ indicates whether model M_m is retrained in cloudlet n_j at time slot t with $y_{m,j,t} = 1$, or not with $y_{m,j,t} = 0$.

The offline version of the user satisfaction maximization problem can be formulated as an integer linear programming (ILP) as follows.

$$\underset{\delta \in \Omega}{\text{Maximize}} \quad \sum_{t=1}^{|\mathbb{T}|} \sum_{r_k \in R_t} \sum_{j=1}^{|N|} U(r_k, n_j, t) \cdot x_{k,j,t} \tag{26}$$

subject to

$$\text{Eq. (4), (7), (13), (14), (15), (16), and (17)}$$

$$\sum_{j=1}^{|N|} y_{m,j,t} \leq 1, \ \forall M_m \in \mathcal{M}, \ \forall \ t \in \mathbb{T} \tag{27}$$

$$\sum_{j=1}^{|N|} x_{k,j,t} + \sum_{j=1}^{|N|} y_{m_k,j,t} \leq 1, \ \forall r_k \in R_t, \ \forall \ t \in \mathbb{T} \tag{28}$$

$$acc_{m_k}(t) \cdot x_{k,j,t} \geq \theta_k, \ \forall r_k \in R_t, \ \forall \ t \in \mathbb{T} \tag{29}$$

$$delay(r_k, n_j, t) \cdot x_{k,j,t} \leq \beta \cdot D_k, \ \forall r_k \in R_t, \ \forall \ t \in \mathbb{T} \tag{30}$$

$$y_{m,j,t} = 0, \ \forall M_m \in \mathcal{M} \setminus \mathcal{M}(t), \ \forall n_j \in N, \ \forall \ t \in \mathbb{T} \tag{31}$$

$$y_{m,j,t} \in \{0, 1\}, \ \forall M_m \in \mathcal{M}, \ \forall n_j \in N, \ \forall \ t \in \mathbb{T} \tag{32}$$

$$x_{k,j,t} \in \{0, 1\}, \ \forall r_k \in R_t, \ \forall n_j \in N, \forall \ t \in \mathbb{T} \tag{33}$$

Constraint (27) ensures that a model is retrained in a cloudlet only if it is chosen for retraining at time slot t. Constraint (28) guarantees that a request will not be admitted if its requested model is in retraining at time slot t. Constraint (29) ensures that the service accuracy requirement of each admitted request of r_k must be met. Constraint (30) ensures that the service delay of each admitted request of r_k is no greater than β times its delay requirement with a constant $\beta > 1$. Constraint (31) ensures that a model M_m will not be retrained if there is no update on its attribute data, i.e., $M_m \notin \mathcal{M}(t)$.

V. Online Algorithm for the User Satisfaction Maximization Problem

In this section, we develop an online algorithm for the user satisfaction maximization problem, by leveraging an effective prediction mechanism and the multi-armed bandit optimization technique. We start with an overview of the proposed online algorithm. We then devise a prediction algorithm, the MAB algorithm, to predict future different types of requests and service model updating. With the prediction results at the previous time slot (or each round pulling in the MAB), we develop an efficient scheduling algorithm that jointly chooses service models for retraining and user requests for services at the current time slot. We finally analyze the time complexity and performance (the expected regret) of the proposed online algorithm.

A. Overview of the Proposed Algorithm

The proposed online algorithm proceeds as follows. It leverages the MAB technique by introducing a number of arms, with each corresponding to a potential resource allocation ratio. At the beginning of each time slot $t \in \mathbb{T}$, the algorithm randomly chooses an arm according to the probabilities $\{p_{\delta,t} \mid \delta \in \Omega\}$ that are calculated based on the historical rewards collected by each arm. It then predicts the number of requests of each service model in $\mathcal{M}(t)$ and their admission rates, by applying a prediction mechanism that analyzes historical traces of past requests in the DT of that model, assuming that there is a DT for each service model in the system that records past requests for the model and their acceptance rate in the past.

Given a chosen ratio δ_t at time slot t, the algorithm chooses models for retraining based on the prediction results, with the given fractional computing resource budget for model retraining. Having chosen models for retraining, it then admits requests requesting services from those non-chosen service models, using the residual computing resource on each cloudlet. It finally calculates the estimated reward $\tilde{\mu}_{\delta,t}$ for each δ and updates the probabilities $\{p_{\delta,t+1} \mid \delta \in \Omega\}$ of each ratio for the next time slot $t+1$.

B. Predictions of Future Requests

If requests for service models can be accurately predicted, model retraining can be scheduled in advance to provide high-fidelity services to the requests in future. We develop an efficient prediction mechanism to predict the number of different types of service requests and their admission rates as follows. Given time slot t, let $|R_{m,t'}|$ be the number of requests for each model $M_m \in \mathcal{M}$ at a previous time slot t' with $t' \leq t$, and all historical request information is kept in the DT, $DT(M_m)$, of requests for model M_m. For the sake of convenience, let $\gamma_{m,t'}$ be the percentage of admitted requests in $R_{m,t'}$ at each previous time slot t' with $t' \leq t$. Similarly, the historical admission rate $\gamma_{m,t'}$ of requests for model M_m is stored in $DT(M_m)$, too, and they are all given by time slot t.

To predict the number of requests for each model M_m in the next consecutive S (≥ 1) time slots starting from time slot $t+1$ to time slot $t+S$ (or the next sliding window S), we here adopt the Long-Short-Term-Memory (LSTM) method [17]. Denote by $\hat{q}_{m,t''}$ the predicted number of requests for model M_m at time slot $t'' \in [t+1, t+S]$. Similarly, denote by $\hat{\gamma}_{m,t''}$ the predicted percentage of admitted requests in $R_{m,t''}$ for each model M_m at time slot $t'' \in [t+1, t+S]$.

The LSTM method models the temporal pattern of request admissions by analyzing the historical traces of requests for service model M_m stored in $DT(M_m)$, which works as a non-linear function $f(.,\ldots,.)$ and outputs future predictions. It first predicts the number $\hat{q}_{m,t+1}$ of requests and the admission rate $\hat{\gamma}_{m,t+1}$ for each model $M_m \in \mathcal{M}$ at time slot $t+1$ as outputs of $f(|R_{m,1}|, \gamma_{m,1}, \ldots, |R_{m,t}|, \gamma_{m,t})$. Then, $\hat{q}_{m,t+2}$ and $\hat{\gamma}_{m,t+2}$ are obtained by feeding the values of $\hat{q}_{m,t+1}$ and $\hat{\gamma}_{m,t+1}$ into the LSTM. This procedure continues until $\hat{q}_{m,t''}$ and $\hat{\gamma}_{m,t''}$ at each time slot $t'' \in [t+1, t+S]$ are obtained. The prediction algorithm is detailed in `Algorithm 1`.

Algorithm 1 Prediction Algorithm for Predicting Requests Requesting for Model M_m in the Future Time Window S Starting at Time Slot $t \in \mathbb{T}$.

Input: The historical traces of requests requesting each model $M_m \in \mathcal{M}$ stored in $DT(M_m)$ until time slot t, and its trained LSTM.

Output: Predict the number of requests $\hat{q}_{m,t''}$ and their admission rate $\hat{\gamma}_{m,t''}$ for model M_m at each time slot t'' with $t'' \in [t+1, t+S]$.

Let $\{|R_{m,t'}| \mid 1 \le t' \le t\}$ be the sequence of the number of requests requesting model M_m at time slot t' with $t' \le t$;
Let $\{\gamma_{m,t'} \mid 1 \le t' \le t\}$ be the admission rate of requests requesting model M_m at time slot t' with $t' \le t$;
Feed historical information $\{|R_{m,t'}|, \gamma_{m,t'} \mid 1 \le t' \le t\}$ to train the LSTM;
Predict the number $\hat{q}_{m,t+1}$ of requests and the admission rate $\hat{\gamma}_{m,t+1}$ of requests for model M_m by utilizing the LSTM;
for $t'' \in [t+2, t+S]$ **do**
    Predict $\hat{q}_{m,t''}$ and $\hat{\gamma}_{m,t''}$ for model M_m at time slot t'' by feeding $\{\hat{q}_{m,t''-1}, \hat{\gamma}_{m,t''-1}\}$ into the LSTM;
end for;
return The predicted number $\hat{q}_{m,t''}$ and admission rate $\hat{\gamma}_{m,t''}$ of requests for model M_m at time slot $t'' \in [t+1, t+S]$.

C. Algorithms for Model Retraining and Request Admissions Under a Given Resource Allocation Policy

To explore non-trivial tradeoffs of allocating limited resource to model retraining and inference services at each time slot t, we start with choosing some models for retraining under a given resource allocation ratio δ_t, where a model is chosen for retraining based on the potential benefit brought by it in future, and the potential profit can be estimated using the prediction mechanism in `Algorithm 1`, which predicts not only the number of requests but also the average admission rate of the requests for each model M_m. In the following, we show how to estimate the potential profit brought by retraining a model in future.

Given time slot t, we calculate the contribution $\hat{U}_{m,t}$ to the optimization objective by admitting requests for model $M_m \in \mathcal{M}$. If model M_m is not chosen for retraining at time slot t, its accuracy is $\widehat{acc}_m(t)$ $(= g\left(\frac{F(M_m,t-1)+1}{|\mathbb{T}|}\right))$, and $\hat{U}_{m,t}$ can be obtained by reducing the problem to the maximum profit GAP. The latter is to pack as many items as possible into bins so that the total profit of packed items is maximized, subject to the capacity on each bin.

There are $|N|$ bins with each corresponding to a cloudlet, and the capacity of bin $j \in N$ is $(1-\delta_t) \cdot C_j$. There are $|R_t|$ items with each corresponding to a request. Request $r_k \in R_t$ can be assigned to bin j if $\widehat{acc}_{m_k}(t) \ge \theta_k$ and its actual end-to-end delay $delay(r_k, n_j, t)$ is no greater than βD_k. The profit brought by this assignment is $profit_2(k,j) = U(r_k, n_j, t)$ and the weight is $weight_2(k,j) = C^{inf}_{m_k}$; otherwise (the accuracy or the delay requirement of request r_k cannot be satisfied), the profit is 0 and the weight is ∞. Notice that request assignment to cloudlets at time slot t can be performed by applying the approximation algorithm [6].

Let $\hat{U}_{m,t}$ be the sum of utilities of admitted requests at time slot t in the solution delivered by the approximation algorithm. We now estimate the profit brought by retraining a model $M_m \in \mathcal{M}(t)$. Since the freshness of each model M_m at time slot $(t-1)$ is given (i.e., $F(M_m, t-1)$), if model M_m is retrained at time slot t, then its service accuracy evolves in a different trajectory in the next S time slots, compared with that without retraining. Let $\Delta acc_m(t')$ be the accuracy gain of model M_m at time slot $t' \in [t+1, t+S]$ after its retraining at time slot t. Then, the potential cumulative profit by model M_m in the next S time slots is

$$\Delta U_{m,t} = \alpha \sum_{t'=t+1}^{t+S} \hat{\gamma}_{m,t'} \cdot \hat{q}_{m,t'} \cdot \Delta acc_m(t') - \hat{U}_{m,t}, \quad (34)$$

where the first term in RHS of Eq. (34) is the total increase of the optimization objective in future time slot interval $[t+1, t+S]$ caused by retraining model M_m at time slot t, $\hat{q}_{m,t'}$ and $\hat{\gamma}_{m,t'}$ are the number of requests and the average admission rate of requests for model M_m at time slot $t' \in [t+1, t+S]$, respectively. The second term $\hat{U}_{m,t}$ is the contribution to the optimization objective by admitting those requests for model M_m if model M_m has not been chosen for retraining at time slot t.

It can be seen that a model M_m will not be chosen for retraining at time slot t if $\Delta U_{m,t} \le 0$, as its retraining at time slot t will not bring any positive profit. Given the computing resource budget $\delta_t \cdot C_j$ on each cloudlet n_j at time slot t, we reduce the model selection problem for retraining at time slot t to the maximum profit GAP too. That is, there are $|N|$ bins with each corresponding to a cloudlet in the MEC network. The capacity on each bin j is the computing resource budget $\delta_t \cdot C_j$ of cloudlet n_j with $1 \le j \le |N|$. There are $|\mathcal{M}(t)|$ items with each corresponding to a service model in $\mathcal{M}(t)$. For a model $M_m \in \mathcal{M}(t)$, it brings a positive potential profit $\Delta U_{m,t} > 0$ if packed to bin j with the residual capacity no less than $C^{train}_{m,t}$, then the profit brought by this assignment is $profit_1(m,j) = \Delta U_{m,t}$ and the weight is $weight_1(m,j) = C^{train}_{m,t}$; otherwise ($\Delta U_{m,t} \le 0$), the profit is $profit_1(m,j) = 0$ and $weight_1(m,j) = \infty$. Let $\mathcal{M}^T(t)$ be the set of chosen models for retraining at time slot t, which is a solution delivered by the approximation algorithm for maximum profit GAP in [6].

Having chosen service models in $\mathcal{M}^T(t)$ for retraining at time slot t, requests in set R'_t $(= \cup_{M_m \in \mathcal{M}^T(t)} R_{m,t})$ cannot be admitted at time slot t, due to the unavailability of their requested models. However, models in $\mathcal{M} \setminus \mathcal{M}^T(t)$ that have not been chosen can continue to provide services to their users at time slot t, in other words, requests in set $R_t \setminus R'_t$ can still be admitted by assigning them to model instances in cloudlets at time slot t. This request assignment can also be reduced to the maximum profit GAP. The reduction is almost identical to the one of choosing models for retraining. Specifically, the request assignment problem is to admit as many requests in $R_t \setminus R'_t$ as possible to maximize the total user satisfaction at time slot t, subject to the residual resource budget $(1-\delta_t) \cdot C_j$ on each cloudlet n_j, where a fraction $(1-\delta_t)$ of computing resource capacity of cloudlet n_j is

allocated as the budget for request admissions at time slot t with $0 < \delta_t < 1$.

The detailed algorithm for jointly choosing service models for retraining and service requests for admissions at time slot t is given in `Algorithm 2`.

D. Online Algorithm

In the following, we devise a Multi-Armed Bandit (MAB) algorithm for the user satisfaction maximization problem, by choosing a resource allocation ratio $\delta \in \Omega$ for each cloudlet. We treat the choice of δ as pulling an arm at each time slot (round), and only one arm is pulled with a certain probability from the pool Ω at each round. Consequently, a reward is obtained through the pulling action, choosing models for retraining, and choosing requests for admissions by `Algorithm 2`. Specifically, if a resource allocation ratio $\delta \in \Omega$ is chosen at time slot t, it results in one resource allocation policy for service models for retraining and user requests for admissions at time slot t, by invoking `Algorithm 2`. The solution delivered by `Algorithm 2` consists of two parts: one is the potential net profit $\Delta U_{m,t}$ in future by choosing a subset of models $M_m \in \mathcal{M}^T(t)$ for retraining; another is to choose a subset of requests for services such that the total user satisfaction on admissions of the chosen requests is $U_{m,t} = \sum_{r_k \in R_{m,t}} \sum_{j=1}^{|N|} U(r_k, n_j, t) \cdot x_{k,j,t}$, where $x_{k,j,t}$ is a binary variable indicating whether request r_k is admitted or not at time slot t. Thus, for any arm (a ratio δ in Ω), let $Reward_{\delta,t}$ be the amount of reward received if the arm (picking δ) is pulled at time slot t. Then, $Reward_{\delta,t} = \sum_{M_m \in \mathcal{M} \setminus \mathcal{M}^T(t)} U_{m,t}$, where $U_{m,t}$ is obtained by `Algorithm 2`.

Let δ_t be the chosen ratio in Ω by the MAB algorithm at each time slot $t \in \mathbb{T}$, and let U be the objective function that is defined as follows.

$$U = \sum_{t=1}^{|\mathbb{T}|} \sum_{r_k \in R_t} \sum_{j=1}^{|N|} U(r_k, n_j, t) \cdot x_{k,j,t}. \tag{35}$$

The relationship between the cumulative reward $\sum_{t=1}^{|\mathbb{T}|} Reward_{\delta_t,t}$ and the objective function is given as follows. By the definition of $U_{m,t}$, $U = \sum_{t=1}^{|\mathbb{T}|} \sum_{M_m \in \mathcal{M} \setminus \mathcal{M}^T(t)} U_{m,t}$. Hence, $\sum_{t=1}^{|\mathbb{T}|} Reward_{\delta_t,t}$ is the value of U delivered by the online algorithm. Without loss of generality, let $\mu_{\delta,t}$ be the normalized value of $Reward_{\delta,t}$ in the interval [0, 1] that is the value of $Reward_{\delta,t}$ divided by $Reward_{\max}$, while $Reward_{\max}$ is an upper bound of the maximum reward that can be derived as follows. By the definition of user satisfaction $U(r_k, n_j, t)$ in Eq. (15), we have $U(r_k, n_j, t) \leq 1$. As $\sum_{j=1}^{|N|} x_{k,j,t} \leq 1$ for each request $r_k \in R_t$, we have $\mu_{\delta,t} \leq |R_t|$. Hence, an upper bound on $Reward_{\delta,t}$ for any $\delta \in \Omega$ at any time slot $t \in \mathbb{T}$ is $Reward_{\max} = \max\{|R_t| \mid t \in \mathbb{T}\}$. We now assume that the amount $\mu_{\delta,t}$ of reward obtained by any ratio $\delta \in \Omega$ at time slot t is normalized.

The expected cumulative regret $\mathbb{E}[Reg]$ of the online algorithm is

$$\mathbb{E}[Reg] = \mathbb{E}\left[\max_{\delta \in \Omega} \sum_{t=1}^{|\mathbb{T}|} \mu_{\delta,t} - \sum_{t=1}^{|\mathbb{T}|} \mu_{\delta_t,t}\right], \tag{36}$$

Algorithm 2 Algorithm for Jointly Choosing Models for Retraining and Request Admissions Under a Given Resource Allocation Ratio δ_t at Time Slot $t \in \mathbb{T}$

Input: An MEC network $G = (N, E)$, a set $\mathcal{M}(t)$ of service models that can be retrained at time slot $t \in \mathbb{T}$, with a potential profit $\Delta U_{m,t}$ and the amount $C_{m,t}^{train}$ of resource consumed for the retraining of model $M_m \in \mathcal{M}(t)$. And a set R_t of requests with each having accuracy and delay requirements.

Output: Maximize the long-term profit that is expressed by the sum of the total profit of retrained models and the total user satisfaction on admitted requests in R_t at time slot t.

/* stage 1: choosing models for retraining */;
for each cloudlet $n_j \in N$ **do**
  Bin j with capacity $\delta_t \cdot C_j$ is cloudlet n_j
end for;
for each model $M_m \in \mathcal{M}(t)$ **do**
  Predict $\hat{q}_{m,t'}$ and $\hat{\gamma}_{m,t'}$ at each time slot t' with $t' \in [t+1, t+W]$, by invoking `Algorithm 1`;
  Calculate $\Delta U_{m,t}$ by Eq. (34);
  for each cloudlet $n_j \in N$ **do**
    if ($\Delta U_{m,t} > 0$) & (the residual computing resource of cloudlet n_j is no less than $C_{m,t}^{train}$) **then**
      $profit_1(m,j) \leftarrow \Delta U_{m,t}$; $weight_1(m,j) \leftarrow C_{m,t}^{train}$, the profit is obtained with the weight assignment, if cloudlet n_j hosts model M_m's retraining
    else
      $profit_1(m,j) \leftarrow 0$; $weight_1(m,j) \leftarrow \infty$
    end if
  end for
end for;
A set $\mathcal{M}^T(t)$ of models for retraining at time slot t is obtained, by applying the approximation algorithm for the maximum-profit GAP in [6];
/* Stage 2: service request admission assignments */;
$R'_t \leftarrow \cup_{M_m \in \mathcal{M}^T(t)} R_{m,t}$; /* requests in R'_t will not be admitted at time slot t */
for each cloudlet $n_j \in N$ **do**
  Bin j with capacity $(1-\delta_t) \cdot C_j$ is cloudlet n_j
end for;
for each request $r_k \in R_t \setminus R'_t$ **do**
  for each cloudlet $n_j \in N$ **do**
    if ($acc_{m_k}(t) \geq \theta_k$) and ($delay(r_k, n_j, t) \leq \beta \cdot D_k$) and (the residual computing resource of cloudlet n_j is no less than $C_{m_k}^{inf}$) **then**
      $profit_2(k,j) \leftarrow U(r_k, n_j, t)$; $weight_2(k,j) \leftarrow C_{m_k}^{inf}$, the profit is obtained with the weight assignment, if request r_k is assigned to cloudlet n_j
    else
      $profit_2(k,j) \leftarrow 0$, $weight_2(k,j) \leftarrow \infty$
    end if
  end for
end for;
Let $R_t^{admit} \subseteq R_t \setminus R'_t$ be the set of admitted requests delivered by the approximation algorithm in [6];
return A feasible solution $(\mathcal{M}^T(t), R_t^{admit}, \delta_t)$ for model retraining and inference services by request admissions at time slot t.

where δ_t is the ratio chosen by the online algorithm at time slot t.

The online algorithm proceeds as follows. At each time slot $t \in \mathbb{T}$, the MAB algorithm randomly pulls an arm δ_t in set Ω with a given probability $p_{\delta_t,t}$ in the probability space $\{p_{\delta,t} \mid \delta \in \Omega\}$, and the probability is computed based on the cumulative estimated reward obtained by the arm so far (from time slot 1 to time slot $t-1$). It then updates the estimated

reward $\tilde{\mu}_{\delta,t}$ of pulling each arm associated with a ratio δ at time slot t by Eq. (37),

$$\tilde{\mu}_{\delta,t} = \frac{\mu_{\delta,t}}{p_{\delta,t}} \cdot \mathbb{I}\{\delta_t = \delta\}, \tag{37}$$

where $\mu_{\delta,t}$ is the actual reward delivered by the choice, and $\mathbb{I}\{\delta_t = \delta\}$ is an indicator function with $\mathbb{I}\{\delta_t = \delta\} = 1$ if $\delta_t = \delta$; $\mathbb{I}\{\delta_t = \delta\} = 0$ otherwise [2]. It finally updates the probabilities $\{p_{\delta,t+1} \mid \delta \in \Omega\}$ for choosing an arm δ at time slot t, based on the cumulative estimated reward $\sum_{t'=1}^{t} \tilde{\mu}_{\delta,t'}$ by Eq. (38).

$$p_{\delta,t+1} = \frac{\exp\left(\eta \sum_{t'=1}^{t} \tilde{\mu}_{\delta,t'}\right)}{\sum_{\delta' \in \Omega} \exp\left(\eta \sum_{t'=1}^{t} \tilde{\mu}_{\delta',t'}\right)} \tag{38}$$

where the learning rate η is real with $0 < \eta < 1$, which will be determined in Theorem 3 later, while $\sum_{t'=1}^{t-1} \tilde{\mu}_{\delta,t'}$ is the cumulative estimated reward brought by choosing ratio δ in the past till time slot t, and $p_{\delta,1} = \frac{1}{|\Omega|}$ initially [2].

Algorithm 3 Online Algorithm for the User Satisfaction Maximization Problem

Input: An MEC network $G = (N, E)$, a set V of physical objects with a deployed DT, DT_i, at cloudlet $h(DT_i)$ for each object v_i, a set $\mathcal{M}$ of service models, a set R_t of service requests at each time slot $t \in \mathbb{T}$, and a set Ω of potential resource allocation ratios.

Output: Maximize the total user satisfaction of admitted requests over time horizon $\mathbb{T}$.

Calculate a shortest path in G for each pair of cloudlets, and the weight of each link is the transmission delay per unit data along the link;
$\mathbb{S} \leftarrow \emptyset$; /* the solution */
Set a learning rate η;
$t \leftarrow 1$;
for each $\delta \in \Omega$ **do**
  Initial the probability $p_{\delta,1}$ with $p_{\delta,1} \leftarrow \frac{1}{|\Omega|}$; $\tilde{\mu}_{\delta,0} \leftarrow 0$;
end for;
while $t \leq |\mathbb{T}|$ **do**
  Select ratio δ_t according to probabilities $\{p_{\delta,t} \mid \delta \in \Omega\}$;
  A feasible solution $(\mathcal{M}^T(t), R_t^{admit}, \delta_t)$ for model retraining and request assignments is obtained at time slot t, by invoking `Algorithm 2` with the chosen δ_t;
  for each $\delta \in \Omega$ **do**
    Update $\tilde{\mu}_{\delta,t} \leftarrow \frac{\mu_{\delta,t}}{p_{\delta,t}} \cdot \mathbb{I}\{\delta_t = \delta\}$;
    Update $p_{\delta,t+1} \leftarrow \frac{\exp\left(\eta \sum_{t'=1}^{t} \tilde{\mu}_{\delta,t'}\right)}{\sum_{\delta' \in \Omega} \exp\left(\eta \sum_{t'=1}^{t} \tilde{\mu}_{\delta',t'}\right)}$
  end for;
  $\mathbb{S} \leftarrow \mathbb{S} \cup \{(\mathcal{M}^T(t), R_t^{admit}, \delta_t)\}$;
  $t \leftarrow t + 1$;
end while;
return A feasible solution $\mathbb{S}$ to the problem.

The detailed MAB algorithm is given in `Algorithm 3`, and its flow chart is given in Figure 2.

E. Algorithm Analysis

Lemma 1: $\tilde{\mu}_{\delta,t}$ is an unbiased estimator of $\mu_{\delta_t,t}$, i.e.,

$$\mathbb{E}[\tilde{\mu}_{\delta,t}] = \mu_{\delta_t,t}. \tag{39}$$

As $\tilde{\mu}_{\delta,t} = \frac{\mu_{\delta,t}}{p_{\delta,t}} \cdot \mathbb{I}\{\delta_t = \delta\}$, $\tilde{\mu}_{\delta,t} = 0$ when $\delta_t \neq \delta$. According to the definition of $p_{\delta,t}$, we have

$$\mathbb{E}[\tilde{\mu}_{\delta,t}] = \frac{\mu_{\delta,t}}{p_{\delta,t}} \cdot \mathbb{I}\{\delta_t = \delta\} \cdot p_{\delta,t} = \frac{\mu_{\delta_t,t}}{p_{\delta_t,t}} \cdot p_{\delta_t,t} = \mu_{\delta_t,t}. \tag{40}$$

The lemma then follows.

Theorem 3: Given an MEC network $G = (N, E)$, a finite time horizon $\mathbb{T}$, a set V of objects with DT_i of each object $v_i \in V$ deployed in cloudlet $h(DT_i)$, a set $\mathcal{M}$ of service models, a set R_t of requests for models in $\mathcal{M}$ at time slot $t \in \mathbb{T}$, an adjustable learning rate η with $0 < \eta \leq 1$, and a fixed set Ω of potential resource allocation ratios, there is an online learning algorithm, `Algorithm 3`, for the user satisfaction maximization problem, which delivers a solution whose expected cumulative regret is no greater than $\eta \cdot |\Omega| \cdot |\mathbb{T}| + \frac{\ln |\Omega|}{\eta}$. Particularly, when $\eta = \sqrt{\frac{\ln |\Omega|}{|\Omega| \cdot |\mathbb{T}|}}$, the expected cumulative regret is $\mathcal{O}(\sqrt{|\mathbb{T}| \cdot |\Omega| \cdot \ln |\Omega|})$. The time complexity of `Algorithm 3` is $\mathcal{O}\left(|N|^3 + \max_{t \in \mathbb{T}} \left\{|\Omega| + |N| \cdot (|R_t| + |\mathcal{M}|) \cdot \log \frac{1}{\epsilon} + \frac{|N|}{\epsilon^4}\right\} \cdot |\mathbb{T}|\right)$, where ϵ is a constant with $0 < \epsilon < 1$.

Following `Algorithm 3`, at each time slot t, it first predicts the number of different types of requests and the average admission rate of each type of requests in the next S consecutive time slots, by invoking `Algorithm 1`. It then computes the potential benefit of choosing a model for retraining at the current time slot based on the prediction results, updates the probability $\{p_{\delta,t} \mid \delta \in \Omega\}$ of each ratio δ in Ω, and randomly chooses one ratio δ_t from Ω. It then chooses model retraining and inference services by invoking `Algorithm 2`, provided the resource of each cloudlet has been allocated to these two activities with the chosen ratio δ_t. The core idea behind the algorithm is to reduce the problem to the maximum profit GAP, and a feasible solution for the latter is obtained by applying the approximation algorithm in [6]. This solution includes which models in $\mathcal{M}(t)$ are chosen for retraining and which requests in R_t are assigned to their requested service models in cloudlets. Since none of the resource capacity on any cloudlet is violated by `Algorithm 3`, the accuracy and delay requirements of each admitted request are met, and the solution delivered by the algorithm thus is feasible. The rest is to show that the expected cumulative regret of the proposed online algorithm is sub-linearly upper-bounded by the length of the time horizon $\mathbb{T}$, which is analyzed as follows. Given time slot t, the probabilities of choosing arms are calculated by Eq. (38). By Lemma 1, we have $\mathbb{E}[\tilde{\mu}_{\delta,t}] = \mu_{\delta_t,t}$. Similarly, we have

$$\begin{aligned}\mathbb{E}[\tilde{\mu}_{\delta,t}^2] &= \left(\frac{\mu_{\delta,t}}{p_{\delta,t}}\right)^2 \cdot \mathbb{I}\{\delta_t = \delta\} \cdot p_{\delta,t} \\ &= \frac{\mu_{\delta_t,t}^2}{p_{\delta_t,t}} \leq \frac{1}{p_{\delta_t,t}}, \text{ since } |\mu_{\delta_t,t}| \leq 1\end{aligned} \tag{41}$$

Taking δ_t as a random variable, we have

$$\mathbb{E}\left[\frac{1}{p_{\delta_t,t}}\right] = \sum_{\delta \in \Omega} \frac{1}{p_{\delta,t}} \cdot p_{\delta,t} = |\Omega|. \tag{42}$$

Then, for any $\delta' \in \Omega$, we have

$$\sum_{t=1}^{|\mathbb{T}|} \mu_{\delta',t} - \sum_{t=1}^{|\mathbb{T}|} \mu_{\delta_t,t} = \sum_{t=1}^{|\mathbb{T}|} \mu_{\delta',t} - \sum_{t=1}^{|\mathbb{T}|} \mathbb{E}[\tilde{\mu}_{\delta,t}], \tag{43}$$

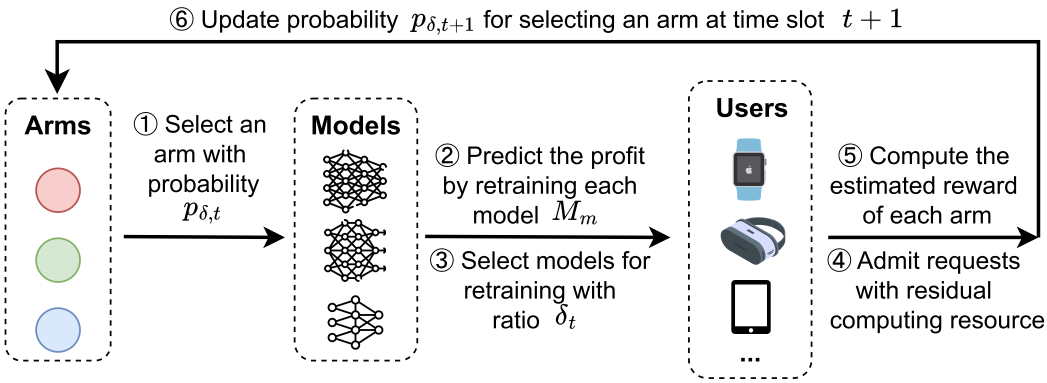


Fig. 2. An illustration of a ratio δ_t chosen for model retraining and inference services at time slot t by the proposed online algorithm.

as $\mathbb{E}[\tilde{\mu}_{\delta,t}] = \mu_{\delta_t,t}$ by Lemma 1. We rewrite the second term in RHS of Eq. (43) as

$$-\mathbb{E}[\tilde{\mu}_{\delta,t}] = \frac{1}{\eta}(\ln \mathbb{E}[\exp(\eta(\tilde{\mu}_{\delta,t} - \mathbb{E}[\tilde{\mu}_{\delta,t}]))] - \ln \mathbb{E}[\exp(\eta\tilde{\mu}_{\delta,t})]), \tag{44}$$

where the first term in RHS of Eq. (44) can be rewritten as

$$\begin{aligned}
&\frac{1}{\eta}\ln \mathbb{E}[\exp(\eta(\tilde{\mu}_{\delta,t} - \mathbb{E}[\tilde{\mu}_{\delta,t}]))] = \frac{1}{\eta}\ln \mathbb{E}[\exp(\eta\tilde{\mu}_{\delta,t} - \eta\mathbb{E}[\tilde{\mu}_{\delta,t}])] \\
&= \frac{1}{\eta}\ln \mathbb{E}[\exp(\eta\tilde{\mu}_{\delta,t})] - \eta\mathbb{E}[\tilde{\mu}_{\delta,t}] \\
&\le \frac{1}{\eta}\mathbb{E}[\exp(\eta\tilde{\mu}_{\delta,t}) - 1 - \eta\tilde{\mu}_{\delta,t}], \text{ since } \ln x \le x - 1 \\
&\le \frac{1}{\eta}\mathbb{E}[\eta^2\tilde{\mu}_{\delta,t}^2], \text{ since } \exp(x) - 1 - x \le x^2 \text{ if } x \in [0,1] \\
&\le \frac{\eta}{p_{\delta_t,t}}, \text{ by (41)}.
\end{aligned} \tag{45}$$

Let $W_t = \frac{1}{\eta}\ln\frac{1}{|\Omega|}\sum_{\delta\in\Omega}\exp\left(\eta\sum_{t'=1}^{t}\tilde{\mu}_{\delta,t'}\right)$. Following the definition of $p_{\delta,t}$ in Eq. (38), the second term in RHS of Eq. (44) can be rewritten as

$$\begin{aligned}
&-\frac{1}{\eta}\ln \mathbb{E}[\exp(\eta\tilde{\mu}_{\delta,t})] = -\frac{1}{\eta}\ln\frac{\sum_{\delta\in\Omega}\exp\left(\eta\sum_{t'=1}^{t}\tilde{\mu}_{\delta,t'}\right)}{\sum_{\delta\in\Omega}\exp\left(\eta\sum_{t'=1}^{t-1}\tilde{\mu}_{\delta,t'}\right)} \\
&= -\frac{1}{\eta}\ln\frac{\sum_{\delta\in\Omega}\exp\left(\eta\sum_{t'=1}^{t-1}\tilde{\mu}_{\delta,t'}\right)\exp(\eta\mu_{\delta,t})}{\sum_{\delta\in\Omega}\exp\left(\eta\cdot\sum_{t'=1}^{t-1}\tilde{\mu}_{\delta,t'}\right)} \\
&= W_{t-1} - W_t,
\end{aligned} \tag{46}$$

where $\sum_{t'=1}^{t}\tilde{\mu}_{\delta,t'}$ is the cumulative estimated reward of ratio δ by time slot t and $W_0 = 0$. Summing up Eq. (44) over time horizon $\mathbb{T}$ and by In eq. (45) and Eq. (46), we have

$$\begin{aligned}
-\sum_{t=1}^{|\mathbb{T}|}\mathbb{E}[\tilde{\mu}_{\delta,t}] &\le \sum_{t=1}^{|\mathbb{T}|}\frac{\eta}{p_{\delta_t,t}} + \sum_{t=1}^{|\mathbb{T}|}(W_{t-1} - W_t) \\
&= \sum_{t=1}^{|\mathbb{T}|}\frac{\eta}{p_{\delta_t,t}} - W_{|\mathbb{T}|}, \text{ as } W_0 = 0.
\end{aligned} \tag{47}$$

By Eq. (42), we have

$$\mathbb{E}\left[\sum_{t=1}^{|\mathbb{T}|}\frac{\eta}{p_{\delta_t,t}}\right] = \eta\cdot|\Omega|\cdot|\mathbb{T}|. \tag{48}$$

By the definition of W_t, we have

$$\begin{aligned}
-W_{|\mathbb{T}|} &= \frac{\ln|\Omega|}{\eta} - \frac{1}{\eta}\ln\left(\sum_{\delta\in\Omega}\exp\left(\eta\sum_{t=1}^{|\mathbb{T}|}\tilde{\mu}_{\delta,t}\right)\right) \\
&\le \frac{\ln|\Omega|}{\eta} - \frac{1}{\eta}\ln\left(\exp\left(\eta\sum_{t=1}^{|\mathbb{T}|}\tilde{\mu}_{\delta',t}\right)\right), \text{ for any } \delta' \in \Omega
\end{aligned} \tag{49}$$

$$= \frac{\ln|\Omega|}{\eta} - \sum_{t=1}^{|\mathbb{T}|}\tilde{\mu}_{\delta',t}, \tag{50}$$

Ineq. (49) holds, due to that $\sum_{\delta\in\Omega}\exp\left(\eta\sum_{t=1}^{|\mathbb{T}|}\tilde{\mu}_{\delta,t}\right) \ge \exp\left(\eta\sum_{t=1}^{|\mathbb{T}|}\tilde{\mu}_{\delta',t}\right)$ for any $\delta' \in \Omega$. We thus have

$$\begin{aligned}
&\mathbb{E}\left[\sum_{t=1}^{|\mathbb{T}|}\mu_{\delta',t} - \sum_{t=1}^{|\mathbb{T}|}\mu_{\delta_t,t}\right] \\
&= \mathbb{E}\left[\sum_{t=1}^{|\mathbb{T}|}\mu_{\delta',t} - \sum_{t=1}^{|\mathbb{T}|}\mathbb{E}[\tilde{\mu}_{\delta,t}]\right], \text{ by Eq. (43)} \\
&\le \mathbb{E}\left[\sum_{t=1}^{|\mathbb{T}|}\mu_{\delta',t} + \sum_{t=1}^{|\mathbb{T}|}\frac{\eta}{p_{\delta_t,t}} - W_{|\mathbb{T}|}\right], \text{ by Eq. (47)} \\
&\le \mathbb{E}\left[\sum_{t=1}^{|\mathbb{T}|}\frac{\eta}{p_{\delta_t,t}} + \frac{\ln|\Omega|}{\eta}\right], \text{ by Eq. (50)} \\
&= \eta\cdot|\Omega|\cdot|\mathbb{T}| + \frac{\ln|\Omega|}{\eta}, \text{ by Eq. (48)}.
\end{aligned} \tag{51}$$

As Ineq. (42) holds for any $\delta' \in \Omega$, by the definition of the expected cumulative regret $\mathbb{E}[Reg]$ in Eq. (36), we have $\mathbb{E}[Reg] \le \eta\cdot|\Omega|\cdot|\mathbb{T}| + \frac{\ln|\Omega|}{\eta}$. When $\eta =$

$\sqrt{\frac{\ln|\Omega|}{|\Omega|\cdot|\mathbb{T}|}}$, the expected cumulative regret is $\mathbb{E}[Reg] = \mathcal{O}(\sqrt{|\mathbb{T}|\cdot|\Omega|\cdot\ln|\Omega|})$.

We finally analyze the time complexity of Algorithm 3. To calculate each user's satisfaction with a service by assigning its request to a cloudlet, this requires finding a shortest path in G between each pair of cloudlets that takes $\mathcal{O}(|N|^3)$ time. At each time slot t, the probability for each ratio in Ω is updated based on the historical information of the network, followed by choosing a ratio δ_t in Ω randomly according to the updated probability, which takes $\mathcal{O}(|\Omega|)$ time. By using the chosen ratio δ_t, the algorithm then invokes Algorithm 2 for computing resource allocation to both model retraining and inference services, which takes $\mathcal{O}\left(|N|\cdot(|R_t|+|\mathcal{M}|)\cdot\log\frac{1}{\epsilon}+\frac{|N|}{\epsilon^4}\right)$ time per time slot. The time complexity of Algorithm 3 thus is $\mathcal{O}\left(|N|^3+\max_{t\in\mathbb{T}}\left(|\Omega|+|N|\cdot(|R_t|+|\mathcal{M}|)\cdot\log\frac{1}{\epsilon}+\frac{|N|}{\epsilon^4}\right)\cdot|\mathbb{T}|\right)$.

VI. Performance Evaluation

In this section, we evaluated the performance of the proposed online algorithm through simulations. Also, we investigated the impacts of important parameters on the performance of the online algorithm.

A. Experimental Environment Settings

Consider an MEC network instance generated by GT-ITM [7], where the number $|N|$ of APs (and their co-located cloudlets) is set at 100 [11]. The bandwidth capacity on each AP is randomly drawn in [100, 200] Mbps [20], and the computing capacity on its co-located cloudlet is randomly drawn from 2,000 to 4,000 MHz [32]. The transmission delay per unit data along a link is randomly drawn in [0.02, 0.05] ms [32]. There are 200 objects (mobile devices) that synchronize with their DTs from 1 to 5 time slots randomly, and the update data size is set within [2,5] MB randomly [11]. There are 100 service models with the number of attributes of each model randomly selected in the range from 5 to 15. The weight $\omega_{m,l}$ of an attribute $attr_{m,l}$ in model M_m is set at $\frac{1}{L_m}$, where L_m is the number of attributes in model M_m. The value of $a>1$ in the submodular function (5) is set at 6. The amount $comp_m$ of computing resource for retraining of model M_m per unit data is randomly drawn in [2, 4] MHz [20]. The amount of computing resource for an inference service is randomly drawn from 200 to 300 MHz [16]. The data processing rate of each model is drawn within [0.5, 2] MB per millisecond [13]. The location of each user is randomly chosen in N, and its requesting service model is randomly drawn in $\mathcal{M}$. The size of input data of each request varies from 1 to 5 MB randomly, the delay requirement of each request is randomly drawn from 10 to 40 ms [16], and the accuracy requirement of each request is randomly drawn in [0.5, 0.8]. The time horizon $\mathbb{T}$ consists of 500 time slots, and there are at most 2,000 requests requesting inference services at any time slot. Parameters α and β are set to 0.5 and 2 in the default setting. The maximum value $\delta_{\max}$ of δ is set to 0.3, and the step size $\frac{\delta_{\max}}{|\Omega|}$ is set to 0.03. Parameter S is set at 10. Unless otherwise specified, we adopt the aforementioned parameters by default.

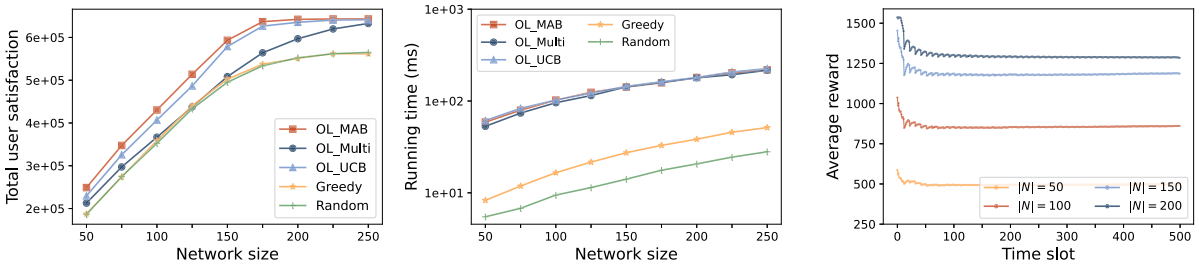
We referred to Algorithm 3 as OL_MAB for short. We proposed four comparison benchmarks against it. The first one is OL_Multi, which randomly chooses a resource allocation ratio δ_t in Ω with a uniform probability $\frac{1}{|\Omega|}$ at each time slot t, and the remaining steps of the algorithm are the same as OL_MAB. The second one is OL_UCB that utilizes the UCB algorithm for adversarial MAB problems, and its exploration parameter is set at 2 [2]. The third one is Greedy, which proceeds as follows. At each time slot, it chooses models for retraining based on the potential benefit brought by each model greedily. It then assigns each chosen model to the cloudlet with the maximum residual resource. This procedure continues until either all chosen models are assigned or there is no adequate computing resource left to meet the resource demand of a chosen model. It finally admits those requests whose requested models have not been chosen for retraining. The last one is Random, which is similar to Greedy except that it randomly chooses models in $\mathcal{M}(t)$ for retraining at each time slot t, and up to 15% of models are chosen ultimately.

The value in each figure is the mean of 30 different network instances, and each network instance is generated by GT-ITM with the same network size [7]. The running time of each algorithm is obtained in a desktop with 11th Gen Intel(R) Core(TM) i7-11700K @ 3.60 GHz, 64G RAM.

B. Performance of the Proposed Online Algorithm

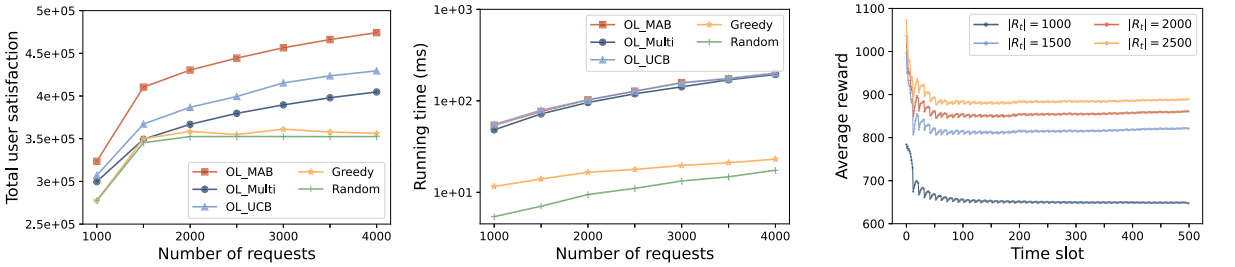
We first evaluated the performance of OL_MAB against the four comparison benchmarks OL_Multi, OL_UCB, Greedy, and Random respectively, by varying network size $|N|$ from 50 to 250, while keeping the number $|R_t|$ of requests at 2,000. Fig. 3 plots the total user satisfaction and running time curves of different algorithms by varying network size $|N|$. It can be seen from Fig.3 that when the network size $|N|$ is set at 50, OL_MAB outperforms OL_Multi and OL_UCB by nearly 17.1% and 10.4%, Greedy and Random by 33%, respectively. When the network size $|N|$ is set at 250, the performances of OL_MAB and OL_UCB are almost identical to that of OL_Multi, since there is more computing resource to support both model retraining and inference services with the increase on the network size.

We then investigated the performance of OL_MAB against other comparison algorithms, by varying the number $|R_t|$ of requests per time slot from 1,000 to 4,000 when network size $|N|$ is fixed at 100. Fig. 4(a) and (b) plot the total user satisfaction and running time curves of different algorithms, respectively. It can be seen from Fig. 4(a) that OL_MAB outperforms OL_Multi and OL_UCB by around 8.1% and 5.4% respectively, when $|R_t|$ is set at 1,000. And OL_MAB outperforms OL_Multi and OL_UCB by nearly 17.2% and 10.4% respectively, when $|R_t|$ is set at 4,000. Fig. 3(b) indicates that the running times of OL_MAB, OL_Multi, and OL_UCB are insignificant. The rationale behind this is that algorithm OL_MAB can better allocate resource between model retraining and inference services. Also, the strategy of choosing models for retraining based on their potential profits in OL_MAB is critical for its super performance. Fig. 3(c) and Fig. 4(c) depict the evolution of average reward obtained by OL_MAB over the number of time slots, by varying both network size $|N|$ and the number $|R_t|$ of requests. Since the MAB technique effectively learns from historical information of the collected rewards, it can make a nearly optimal decision with the optimal reward obtained at each time slot.



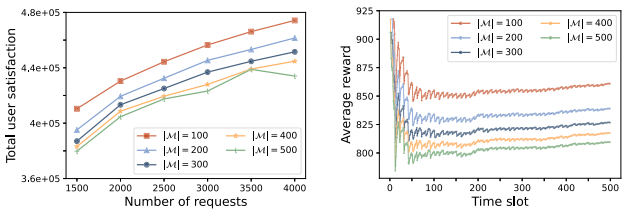
(a) The total user satisfaction (b) The running time (c) The average reward at each time slot

Fig. 3. Performance of online algorithm OL_MAB against the other four comparison online algorithms for the user satisfaction maximization problem, by varying the network size $|N|$ when $|\mathbb{T}| = 500$.



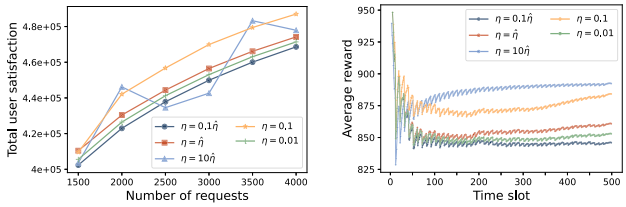
(a) The total user satisfaction (b) The running time (c) The average reward at each time slot

Fig. 4. Performance of online algorithm OL_MAB against the other four comparison algorithms for the user satisfaction maximization problem, by varying the number $|R_t|$ of requests per time slot, when $|\mathbb{T}| = 500$.



(a) The total user satisfaction (b) The average reward

Fig. 5. Performance of Algorithm 3 by varying the number $|\mathcal{M}|$ of models and the number $|R_t|$ of requests per time slot.



(a) The total user satisfaction (b) The average reward

Fig. 6. Performance of Algorithm 3 by varying the learning rate η and the number $|R_t|$ of requests per time slot t.

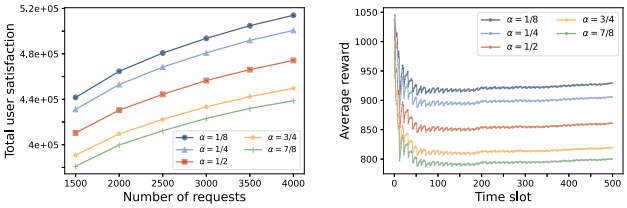
C. Impacts of Parameters on the Performance of the Proposed Online Algorithm

We then investigated the impacts of important parameters of Algorithm 3 on its performance when fixing network size $|N|$ at 100. The parameters include the number $|\mathcal{M}|$ of service models, the value of the learning rate η, the value of coefficient α, the weight $\omega_{m,l}$ for the lth attribute of each model M_m, the maximum value $\delta_{\max}$ of δ, and the number $|\Omega|$ of potential resource allocation ratios in set Ω.

The impact of the number $|\mathcal{M}|$ of models on the performance of OL_MAB is evaluated in Fig. 5 by varying $|\mathcal{M}|$ from 100 to 500. It can be seen from Fig. 5(a) that OL_MAB with $|\mathcal{M}| = 100$ outperforms itself with $|\mathcal{M}| = 500$ by nearly 9.5% when the number $|R_t|$ of requests is set at 4,000 per time slot. With the increase on $|\mathcal{M}|$, more and more models will not be chosen for retraining due to limited resource, resulting in less accurate results from these untrained models. Fig. 5(b) plots the average reward curves delivered by OL_MAB, from which it can be seen that the reward curves become stable with time, because the MAB technique can make a better decision with more historical information available.

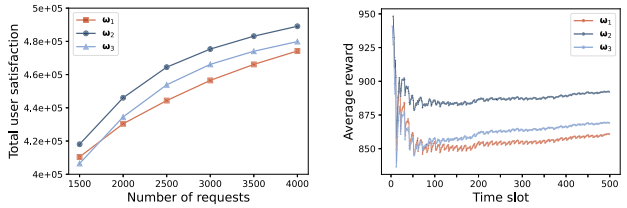
The impact of the learning rate η on the performance of OL_MAB is illustrated in Fig. 6 by setting η at $0.1\hat{\eta}$, $\hat{\eta}$, $10\hat{\eta}$, 0.1, and 0.01, respectively, where $\hat{\eta} = \sqrt{\frac{\ln|\Omega|}{|\Omega|\cdot|\mathbb{T}|}}$ is the best choice by Theorem 3, and $\hat{\eta} \approx 0.0215$ by the default setting. It can be seen from Fig. 6(a) that the performance of OL_MAB with $\eta = 0.1\hat{\eta}$ is around 96% of itself with $\eta = 0.1$ when the number of requests per time slot varies from 1,500 to 4,000. It can be observed that $\eta = 0.1$ is the best choice in most cases, while the performance of OL_MAB with $\eta = 10\hat{\eta}$ fluctuates dramatically with changing numbers of requests. The rationale behind this is that although the value of $\hat{\eta}$ is the best choice by Theorem 3 to minimize the upper bound on the expected cumulative regret $\mathbb{E}[Reg]$, it may not deliver the best performance.

The impact of parameter α on the performance of OL_MAB is given in Fig. 7 by fixing the value of α at 1/8, 1/4, 1/2, 3/4, and 7/8, respectively. It can be seen from Fig. 7(a) OL_MAB with $\alpha = 1/8$ outperforms itself with $\alpha = 7/8$ by nearly 17.2% when the number of requests is set at 4,000 per time slot. This implies that although the user satisfaction with his admitted request is determined by both the service accuracy and service delay, which are normalized to [0, 1], there is a discrepancy between the accuracy and delay requirements of



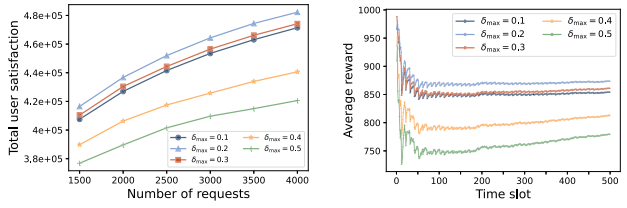
(a) The total user satisfaction (b) The average reward

Fig. 7. Performance of Algorithm 3 by varying α and the number $|R_t|$ of requests per time slot t.



(a) The total user satisfaction (b) The average reward

Fig. 8. Performance of Algorithm 3 by varying attribute weights $\{\omega_{m,l}\}$ and the number $|R_t|$ of requests per time slot t.



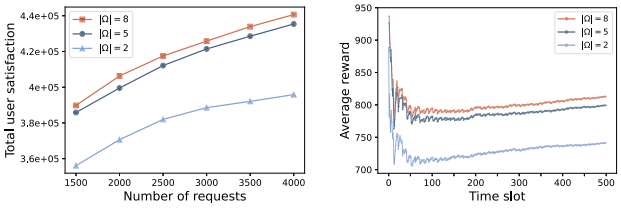
(a) The total user satisfaction (b) The average reward

Fig. 9. Performance of Algorithm 3 by varying $\delta_{\max}$ of δ in Ω and the number $|R_t|$ of requests per time slot t.

the request, and the accuracy requirement is more sensitive than the delay requirement.

The impact of weight $\omega_{m,l}$ of each attribute $attr_{m,l}$ of model M_m on the performance of OL_MAB is plotted in Fig. 8, where three different weight assignments are considered for each model $M_m \in \mathcal{M}$: the first is the equal weight assignment $\boldsymbol{\omega}_1$, i.e., the weight of each attribute in M_m is $\frac{1}{|A_m|}$, the second is a geometric weight assignment $\boldsymbol{\omega}_2$, i.e., the set of weights is $\left\{1/2, 1/4, \ldots, 1-\sum_{i=1}^{|A_m|-1} 1/2^i\right\}$, and the last is a randomly weight assignment $\boldsymbol{\omega}_3$, which assigns each attribute in M_m with a random value between 0 and 1 such that the sum of all attribute weights is 1. It can be seen from Fig. 8(a) that OL_MAB with $\boldsymbol{\omega}_2$ outperforms itself with $\boldsymbol{\omega}_1$ by merely 3.1% when the number of requests is set at 4,000 per time slot. This indicates that the impact of attribute weights on the performance of OL_MAB is insignificant, and the service accuracy of a model is not determined by its attribute weights only, but rather determined by many factors, including its retraining frequency, the weight of each of its attributes, and the data update volume and frequency of each of its attributes.

The impact of parameter $\delta_{\max}$ on the performance of OL_MAB is shown in Fig. 9 by varying $\delta_{\max}$ from 0.1 to 0.5 while fixing the step size $\frac{\delta_{\max}}{|\Omega|}$ at 0.03. It can be seen from Fig. 9(a) that OL_MAB with $\delta_{\max} = 0.2$ outperforms itself with $\delta_{\max} = 0.5$ by nearly 14.7% when the number of requests is set at 4,000 per time slot.



(a) The total user satisfaction (b) The average reward

Fig. 10. Performance of Algorithm 3 by varying the number $|\Omega|$ of ratios and the number $|R_t|$ of requests per time slot t.

The impact of the number $|\Omega|$ of resource allocation ratios on the performance of OL_MAB is evaluated in Fig. 10, by varying $|\Omega|$ from 2 to 8 while fixing the value of $\delta_{\max}$ at 0.4 with step size varying from 0.05 to 0.2. It can be seen from Fig. 10(a) that when the number of requests is set at 4,000 per time slot, the performance of OL_MAB with $|\Omega| = 2$ is around 89.8% of itself with $|\Omega| = 8$. This is because the larger the value of $|\Omega|$, the more precise resource allocation ratios the OL_MAB can deliver, thereby the better utilization of computing resource.

VII. Conclusion and Future Work

In this paper, we investigated fidelity-aware, delay-sensitive inference services in a DT-assisted MEC network within a given time horizon. We first formulated a novel user satisfaction maximization problem with the aim to maximize the cumulative user satisfaction on services. We formulated an ILP solution to the offline version of the problem. We then devised an online algorithm with bounded expected regret for the problem, by leveraging an efficient prediction mechanism and the multi-armed bandit optimization. We finally evaluated the performance of the proposed online algorithm through simulations. Simulation results demonstrate that the proposed algorithm is promising, outperforming its comparison counterparts.

The potential research built upon this study in the future includes investigating other prediction mechanisms, such as deep reinforcement learning, improving resource allocation robustness, and integrating privacy-preserving approaches for updating data used for model retraining, to ensure secure and privacy-aware inference services in DT-assisted MEC networks.

Acknowledgment

The authors appreciate the five anonymous referees and the associate editor for their constructive comments and invaluable suggestions, which helped them to improve the quality and presentation of the article greatly.

References

[1] X. Ai, W. Liang, Y. Zhang, and W. Xu, "Fidelity-aware inference services in DT-assisted edge computing via service model retraining," *IEEE Trans. Services Comput.*, vol. 18, no. 4, pp. 2089–2102, Jul. 2025.

[2] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, nos. 2–3, pp. 235–256, May 2002.

[3] H. B. Beytur, A. G. Aydin, G. de Veciana, and H. Vikalo, "Optimization of offloading policies for accuracy-delay tradeoffs in hierarchical inference," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, May 2024, pp. 1989–1998.

[4] H. Cai, Z. Zhou, and Q. Huang, "Online resource allocation for edge intelligence with colocated model retraining and inference," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, May 2024, pp. 1900–1909.
[5] H. Chen, T. D. Todd, D. Zhao, and G. Karakostas, "Digital twin model selection for feature accuracy," *IEEE Internet Things J.*, vol. 11, no. 7, pp. 11415–11426, Apr. 2024.
[6] R. Cohen, L. Katzir, and D. Raz, "An efficient approximation for the generalized assignment problem," *Inf. Process. Lett.*, vol. 100, no. 4, pp. 162–166, Nov. 2006.
[7] (2025). *GT-ITM*. [Online]. Available: http://www.cc.gatech.edu/projects/gtitm/
[8] Y. Han et al., "A dynamic hierarchical framework for IoT-assisted digital twin synchronization in the metaverse," *IEEE Internet Things J.*, vol. 10, no. 1, pp. 268–284, Jan. 2023.
[9] I. Kadota, A. Sinha, and E. Modiano, "Optimizing age of information in wireless networks with throughput constraints," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2018, pp. 1844–1852.
[10] Y. Kong, P. Yang, and Y. Cheng, "Adaptive on-device model update for responsive video analytics in adverse environments," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 35, no. 1, pp. 857–873, Jan. 2025.
[11] J. Li et al., "AoI-aware user service satisfaction enhancement in digital twin-empowered edge computing," *IEEE/ACM Trans. Netw.*, vol. 32, no. 2, pp. 1677–1690, Apr. 2024.
[12] J. Li et al., "AoI-aware, digital twin-empowered IoT query services in mobile edge computing," *IEEE/ACM Trans. Netw.*, vol. 32, no. 4, pp. 3636–3650, Aug. 2024.
[13] J. Li et al., "Mobility-aware utility maximization in digital twin-enabled serverless edge computing," *IEEE Trans. Comput.*, vol. 73, no. 7, pp. 1837–1851, Jul. 2024.
[14] J. Li et al., "Digital twin-enabled service provisioning in edge computing via continual learning," *IEEE Trans. Mobile Comput.*, vol. 23, no. 6, pp. 7335–7350, Jun. 2024.
[15] J. Li, W. Liang, J. Wang, and X. Jia, "Accumulative fidelity maximization of inference services in DT-assisted edge computing," in *Proc. Int. Conf. Meta Comput. (ICMC)*, Jun. 2024, pp. 64–73.
[16] J. Li et al., "Maximizing user service satisfaction for delay-sensitive IoT applications in edge computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 5, pp. 1199–1212, May 2022.
[17] Y. Li, W. Liang, Z. Xu, W. Xu, and X. Jia, "Budget-constrained digital twin synchronization and its application on fidelity-aware queries in edge computing," *IEEE Trans. Mobile Comput.*, vol. 24, no. 1, pp. 165–182, Jan. 2025.
[18] J. Li, J. Wang, W. Liang, X. Jia, and A. Y. Zomaya, "Inference service fidelity maximization in DT-assisted edge computing," *IEEE Trans. Mobile Comput.*, early access, Aug. 19, 2025, doi: 10.1109/TMC.2025.3600390.
[19] Z. Li et al., "Train large, then compress: Rethinking model size for efficient training and inference of transformers," in *Proc. ICML*, 2022, pp. 5958–5968.
[20] X. Liang, W. Liang, Z. Xu, Y. Zhang, and X. Jia, "Multiple service model refreshments in digital twin-empowered edge computing," *IEEE Trans. Services Comput.*, vol. 17, no. 5, pp. 2672–2686, Sep. 2024.
[21] H. Liu, S. Liu, S. Long, Q. Deng, and Z. Li, "Joint optimization of model deployment for freshness-sensitive task assignment in edge intelligence," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, May 2024, pp. 1751–1760.
[22] S. Liu, D. Wen, D. Li, Q. Chen, G. Zhu, and Y. Shi, "Energy-efficient optimal mode selection for edge AI inference via integrated sensing-communication-computation," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 14248–14262, Dec. 2024.
[23] Y. Ma, W. Liang, J. Li, X. Jia, and S. Guo, "Mobility-aware and delay-sensitive service provisioning in mobile edge-cloud networks," *IEEE Trans. Mobile Comput.*, vol. 21, no. 1, pp. 196–210, Jan. 2022.
[24] X. Ren, Q. Li, H. Du, H. Yao, C. Qiu, and D. Niyato, "Tri-ring: Asynchronous service provisioning with online learning in edge cloud networks," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, May 2025, pp. 1–10.
[25] Y. Shu, Z. Wang, H. Liao, Z. Zhou, N. Nasser, and M. Imran, "Age-of-information-aware digital twin assisted resource management for distributed energy scheduling," in *Proc. GLOBECOM IEEE Global Commun. Conf.*, Dec. 2022, pp. 5705–5710.
[26] A. Slivkins, "Introduction to multi-armed bandits," *Found. Trends Mach. Learn.*, vol. 12, nos. 1–2, pp. 1–286, 2019.
[27] Z. Wang, D. Jiang, and S. Mumtaz, "Network-wide data collection based on in-band network telemetry for digital twin networks," *IEEE Trans. Mobile Comput.*, vol. 24, no. 1, pp. 86–101, Jan. 2025.
[28] D. Wen et al., "Task-oriented sensing, computation, and communication integration for multi-device edge AI," *IEEE Trans. Wireless Commun.*, vol. 23, no. 3, pp. 2486–2502, Mar. 2024.
[29] Z. Xu et al., "Energy or accuracy? Near-optimal user selection and aggregator placement for federated learning in MEC," *IEEE Trans. Mobile Comput.*, vol. 23, no. 3, pp. 2470–2485, Mar. 2024.
[30] Y. Yang et al., "Dynamic human digital twin deployment at the edge for task execution: A two-timescale accuracy-aware online optimization," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 12262–12279, Dec. 2024.
[31] Y. Zeng, R. Zhou, L. Jiao, Z. Han, J. Yu, and Y. Ma, "Efficient online DNN inference with continuous learning in edge computing," in *Proc. IEEE/ACM 32nd Int. Symp. Quality Service (IWQoS)*, Jun. 2024, pp. 1–10.
[32] Y. Zhang, W. Liang, Z. Xu, and X. Jia, "Mobility-aware service provisioning in edge computing via digital twin replica placements," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 11295–11311, Dec. 2024.
[33] Y. Zhang, W. Liang, Z. Xu, W. Xu, and M. Chen, "AoI-aware inference services in edge computing via digital twin network slicing," *IEEE Trans. Services Comput.*, vol. 17, no. 6, pp. 3154–3170, Nov. 2024.
[34] L. Zhao et al., "Adaptive swarm intelligent offloading based on digital twin-assisted prediction in VEC," *IEEE Trans. Mobile Comput.*, vol. 23, no. 8, pp. 8158–8174, Aug. 2024.
[35] X. Zhuang, J. Wu, H. Wu, T. Zhang, and L. Gao, "Joint optimization of model inferencing and task offloading for MEC-empowered large vision model services," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, May 2025, pp. 1–10.

Xuan Ai received the B.S. degree in statistics from the University of Science and Technology of China in 2022. She is currently pursuing the Ph.D. degree with the City University of Hong Kong. Her research interests include cloud and edge computing, serverless computing, and edge intelligence.

Weifa Liang (Senior Member, IEEE) received the B.Sc. degree in computer science from Wuhan University, China, in 1984, the M.E. degree in computer science from the University of Science and Technology of China in 1989, and the Ph.D. degree in computer science from The Australian National University in 1998. He is currently a Full Professor with the Department of Computer Science, City University of Hong Kong. Prior to that, he was a Full Professor at The Australian National University. His research interests include the design and analysis of energy-efficient routing protocols for wireless ad hoc and sensor networks, mobile edge computing (MEC), network function virtualization (NFV), the Internet of Things, digital twins, the design and analysis of parallel and distributed algorithms, approximation algorithms, and graph theory. He serves as an Editor for IEEE TRANSACTIONS ON COMMUNICATIONS.

Caiyi Liu received the B.Sc. degree in electronic information from Xidian University, China, in 2020, and the M.E. degree in information and communication engineering from Beihang University, China, in 2023. She is currently pursuing the Ph.D. degree with the City University of Hong Kong. Her research interests include cloud and edge computing, serverless computing, and edge intelligence.