Truthful Deep Mechanism Design for Revenue-Maximization in Edge Computing With Budget Constraints

Gang Li, *Student Member, IEEE*, Jun Cai, *Senior Member, IEEE*, and Shuang Ni

***Abstract*—In this paper, collaborative task offloading in edge computing is studied, where computation requesters can offload tasks to not only the edge server, but also nearby smartphone users. By considering the fact that smartphone users may not always be willing to provide such computation service because of the consumption of their own energy and resources, a truthful mechanism is designed to provide incentive to smartphone users. The design aims to maximize the net revenue of the service provider and addresses more practical, but more complicated, scenarios of unknown *a prior* distribution information on smartphone users' private information. To tackle this high computational complexity, which makes the traditional mechanism design methods infeasible, a new approach, called truthful deep mechanism, is proposed by leveraging a multi-task machine learning model, where inherently inter-connected collaborator selection and pricing policy determination are decided by designing two deep neural networks. The numerical results show that the proposed deep truthful mechanism can ensure a convergence to a stable state and can satisfy all required economical properties, including individual rationality, incentive compatibility, and budget balance.**

***Index Terms*—Edge computing, deep learning, incentive mechanism, revenue maximization.**

I. Introduction

RECENTLY, in order to further improve smartphone users' Quality of Experience (QoE) and rationally utilize computational resources in Edge Computing (EC), collaborative offloading has been proposed [1], [2], in which smartphone users are allowed to offload tasks to idle terminals in their neighborhood for computation, called collaborators. However, these potential collaborators may not always be willing to provide services without reimbursements since they need to consume their own energy and computation resources. Moreover, in order to better manage computational resources for smartphone users, smartphone users' private information is needed, which is usually unknown to the central controller, e.g., the base station (BS). Thus, incentive mechanisms should be designed to not only incentivize collaborators for participation, but also force smartphone users to truthfully report their private information. Furthermore, in practice, the constraints of budget at smartphone users have to be considered to prevent them from running out of their pockets.

In the incentive mechanism design area, two different objectives [3] are commonly considered. One is social welfare maximization, which maximizes the summation of all participators' utilities, and the other is to maximize the revenue of the service provider. In the literature, compared to works [4]–[6], which aimed to design incentive mechanisms by maximizing the social welfare of the system, fewer studies considered the revenue maximization due to its high complexity in solution [7]. However, the consideration of revenue maximization from the service provider's point of view is also important and meaningful because this reflects the benefit or profit that the service provider earns. Moreover, most existing works, such as [8]–[12], on revenue maximization mechanism design were under the constraint of Bayesian incentive compatibility (BIC), while we consider a dominate-strategy incentive compatibility (DSIC) mechanism design, which is a more stronger condition than BIC. In addition, some work [13], [14] on revenue maximization mechanism design in wireless networks assumed the availability of *a priori* distributions of smartphone users' private information. However, such an assumption may not always be true for practical EC applications. Furthermore, even though work in [15], [16] didn't need *a priori* distributions to design incentive mechanisms with budget balance, their objectives were actually not to maximize the revenue. Therefore, it is necessary to redesign revenue maximization incentive mechanisms by considering unknown *a priori* distributions and at the same time satisfying some economical properties, such as individual rationality, dominate-strategy incentive compatibility, and budget balance. However, such design is very challenging because of the following reasons.

- Traditionally, the designed mechanism should consist of an allocation rule and a pricing policy, where the allocation rule stipulates collaborator selection and the pricing policy

Manuscript received March 13, 2021; revised August 1, 2021; accepted October 26, 2021. Date of publication November 18, 2021; date of current version January 20, 2022. This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) under Discovery and CRD Grants, Concordia University PERFORM RESEARCH Chair Program. The review of this article was coordinated by Dr. Yuanxiong Guo. *(Corresponding author: Jun Cai.)*

Gang Li is with the School of Electronic and Electrical Engineering, Wuhan Textile University, Wuhan 430200, China, and also with the Network Intelligence and Innovation Lab (NI2L), Department of Electrical and Computer Engineering, Concordia University, Montreal, QC H4B 1R6, Canada (e-mail: gang.li@mail.concordia.ca).

Jun Cai is with the Network Intelligence and Innovation Lab (NI2L), Department of Electrical and Computer Engineering, Concordia University, Montreal, QC H4B 1R6, Canada (e-mail: jun.cai@concordia.ca).

Shuang Ni is with MILA, Department of Computer Science, University of Montreal, Montreal, QC H3C 3J7, Canada (e-mail: shuang.ni@mail.concordia.ca).

Digital Object Identifier 10.1109/TVT.2021.3129098

determines how much smartphone users need to pay. In the revenue maximization mechanism design problem, these two aspects are tightly coupled. Existing work [8]–[12] overcome this issue under the constraint of BIC. However, even if the *prior* information is known, it is still unexplored and extremely challenging to design an DSIC mechanism under the constraint of budget balance for edge computing systems;

- Without*a priori* distributions of smartphone users' private information, the traditional solution methods for revenue maximization in the literature become infeasible. Even if there exist *prior* free mechanism design methods, such as [17], [18], they designed the mechanisms under the BIC condition and without considering budget balance constraints. Therefore, a new approach has to be developed to address such complicate mechanism designs;
- The net revenue of the service provider is defined as the sum of all payments from smartphone users minus its costs. Thus, the designed mechanism needs to collect reimbursements as many as possible while still keeping the constraint of budget balance at each smartphone user. These two requirements are contradictory with each other and further complex our problem;
- In order to better assign collaborators, the designed mechanism should force smartphone users to report their true private information. Meanwhile, smartphone users who have tasks to offload may confront more than one idle collaborator available in their proximity, which results in a multi-dimensional bidding setting. Since most existing work on revenue maximization focused on just one- or two-dimensional bidding, those solutions cannot be directly applied to our case.

In light of the above challenges and difficulties, and motivated by the recent advance in deep learning, in this paper, we aim to design a new incentive mechanism integrating deep learning approach, i.e., truthful deep mechanism design, for collaborative task offloading in an EC system. Our objective is to maximize the net revenue of the service provider while considering economical properties in terms of individual rationality, incentive compatibility, and budget balance. Because of the high computational complexity involved, which prevents the application of traditional mechanism design methods, new approach based on a multi-task machine learning model is devised to generate results for two inherently interconnected tasks, i.e., collaborator selection and pricing rule, simultaneously. Specifically, our proposed multi-task machine learning model consists of two related well-designed deep neural networks for collaborator selection and pricing rule, respectively. Then, the outcomes of both deep neural networks jointly determine a loss function, which is the optimization objective of our multi-task machine learning model. Finally, this devised multi-task machine learning model is trained under this defined loss function. The main contributions of this paper are summarized as follows:

- Different from the existing work which focused on social welfare maximization, our work focuses on the more challenging revenue maximization problem with budget balance. What's more, existing revenue maximization mechanisms can only be applied to a fixed bidding valuation distribution, while our proposed mechanism can be suitable for any distribution;
- Unlike [19], which considered one dimensional bidding setting and chose only one winner for each running time, a more general scenario where each smartphone user can submit multiple biddings and multiple winners can be selected in each round is considered to design a truthful mechanism;
- Inspired by the multi-task machine learning model, a truthful deep mechanism is devised to fit our specific problem, and we further evaluate the performance of our proposed incentive mechanism through comprehensive numerical simulations.

The remainder of this paper is organized as follows. Section II is the related works, while the system model is described in Section III. In Section IV, the truthful mechanism framework integrating deep learning is presented. Numerical results are presented in Section V, followed by concluding remarks in Section VI. In the sequel of this paper, $\mathbb{E}\{x\}$ denotes the expectation of the random variable x, $|\mathcal{X}|$ denotes the cardinality of set $\mathcal{X}$, and the superscript H denotes the transpose of a vector.

II. Related Works

The upsurge of studying task offloading in the future evolution of wireless networks was spurring heated discussion on both industrials and academia in past years. Previous efforts on task offloading mainly focused on communication and computational resources allocation [20]–[22], while ignoring the issue of incentivization to computational service providers. Recently, more and more works such as [2], [4], [23]–[24] are shifting their focuses to design incentive mechanisms in edge computing systems. Specifically, with the consideration of network dynamics in edge computing networks, such as different arriving time of tasks, [2] proposed an online incentive mechanism for the collaborative task offloading scenario where an optimization problem was solved to maximize the total social welfare in an online fashion. Furthermore, authors in [4] proposed a multi-tiered edge computing architecture in which an optimization problem was formulated to maximize the profit of the service provider. Then, an incentive mechanism was designed to address the optimization problem. Moreover, in order to ensure the secure trading and resource allocation between smartphone users and edge servers in edge computing, a blockchain-driven resource allocation problem was modelled as a double truthful mechanism to maximize the system efficiency in [23]. In IoT applications [24], by joint consideration of Lyapunov method and the framework of Vickrey-Clarke-Groves (VCG) mechanism, an online truthful mechanism was designed to maximize the long-term social welfare.

Artificial intelligence (AI) is envisioned as a promising technology to be employed in future wireless networks because of its flexibility and ability for solving large-scale problems [25]. AI is a broader concept of machine learning, which is probably the most popular application of AI. Moreover, it has been shown by many researches [26], [27] that compared to conventional methods, the utilization of machine learning can improve network performance. As such, a bunch of works [28]–[34] are starting to design computation resource allocation algorithms in edge computing networks based on machine learning techniques. For instance, a multi-user multi-edge-server computation offloading was formulated in [28] as a non-cooperative potential game where each mobile user maximized its obtained computation resource and reduced its energy consumption. This complex problem was solved by a model-free reinforcement learning technique [29]–[33]. Through applying deep learning for data analysis in an edge computing system, paper in [30] designed a convolutional neural network to collaboratively perform image recognition between edge servers and the cloud in the proposed system. Furthermore, by integrating deep learning and reinforcement learning techniques, [31]–[33] proposed deep reinforcement learning approaches in edge computing networks to maximize the long-term benefits of the whole system. Moreover, in [34], in order to obtain better offloading decisions, task execution time was firstly predicted by a low-rank learning model in the edge computing system. Then, the task offloading problem with predicted execution time was formulated to maximize the number of offloaded tasks.

Notice that the above-mentioned works in edge computing systems only considered either network economical aspects to maximize total social welfare or computation resource allocation by machine learning technology. Different from all of above works, without knowing a prior distribution information on smartphone users' private information, we design a truthful deep mechanism to maximize the total net revenue of the service provider with the consideration of network economical properties in the collaborative edge computing system.

III. System Model and Problem Formulation

In this section, the system model is first introduced. After that we formulate the interactions between the BS and requesters, i.e., smartphone users, as an incentive (truthful) mechanism design problem.

A. System Description

Similar to [1], [2], we consider an EC system, as shown in Fig. 1. In the system, there is a base station (BS) equipped with an EC server. Smartphone users are classified into two categories. $|\mathcal{N}| = N$ users are called requesters, who would like to offload computation tasks, while the rest of $|\mathcal{M}| = M$ smartphone users are collaborators, who are recruited by the BS at the beginning of offloading process and willing to provide computation service if rewards are provided. The computation tasks can come from the applications, such as data compression, face recognition, and virus scan. Each collaborator can serve at most one task at any time because of constraints on its computation capacity and the resulted processing delay. In this paper, we use the term "task executor" to represent either the collaborator or the BS, and each requester can only offload its task to at most one task executor. In the considered system model, each requester first submits its preference value, requested computation resource, available budget, and available collaborators[1] to the BS, who then assigns

[1]Note that the available collaborators nearby requester i can be found by applying the discovery approach [37].

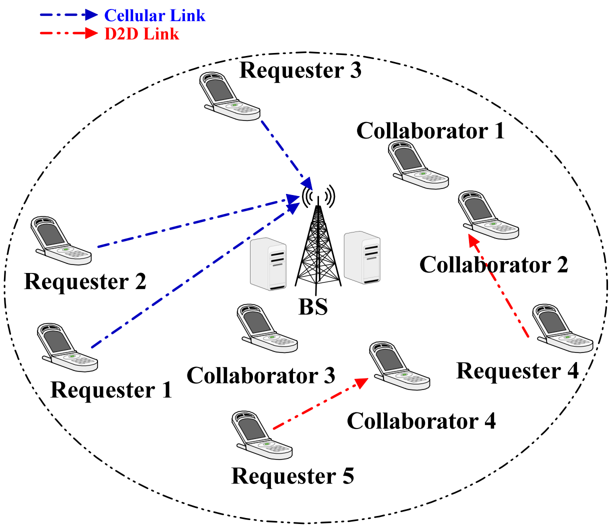

Fig. 1. Collaborative task offloading in edge computing.

task executors to complete all requested tasks. After completion, the task executors send results to the corresponding requesters, and the requesters should make compensation to the corresponding task executors, both of which could be done through D2D links [1] or cellular links. Note that if computational results are not received successfully by the corresponding requester, no reimbursement will be given to the task executor. Therefore, collaborators will try their best to complete the offloading process.[2]

Obviously, such interactions between the task executors and the requesters can be modelled as a truthful mechanism design problem, where the requesters are buyers and the task executors are sellers. In this truthful mechanism design, for buyer $i \in \mathcal{N}$, its submitted bid can be denoted as $b_i = \{b_{i,1}, b_{i,2}, \cdots b_{i,M+1}, f_i, B_i\}$, where $b_{i,j}$ is the submitted valuation of buyer i for task executor j, f_i denotes the corresponding requested computation frequency, and B_i denotes the budget of smartphone user i. Note that in practice, smartphone users cannot run out of their pockets to reimburse selected task executors. Thus, the budget balance means the payment made by any smartphone user cannot be more than a fixed upper limit, i.e., B_i. We further define a matrix $\boldsymbol{b} = \{b_1; b_2; \cdots; b_N\}$, which is composed of all requesters' bids, and denote $\boldsymbol{v}$ as the truthful bids of $\boldsymbol{b}$. As previously stated, it is practical that $\boldsymbol{b}$ are private known information to requesters themselves and may vary with the time. Therefore, similar to works in [14], [35], [36], the bid of smartphone user i follows a distribution F_i. But, unlike those work, which always assumed the availability of this *a prior* distribution, we consider a more general case where this distribution may be unavailable. Moreover, let l_i be the task size in bit unit of requester i, and the index $M+1$ represents the BS and $\mathcal{M}_1 = \{M+1 \cup \mathcal{M}\}$. Note that requesters can have different values of $b_{i,j}$ to different task executors. For example,

[2]If the collaborator moves out of the communication coverage of requesters just after the start of offloading process, the task will probably not be completed because of transmission interruption, which incurs no rewards to collaborators. Therefore, there are no incentives for collaborators to move out. For the movement of requester, the computation result and the reimbursements can also be received and transmitted through the cellular link even if requesters move out of the cell.

a requester with a latency-intensive task may value more to collaborators who have better channel conditions, while the requester with a computation-intensive task may value more to the BS.

Our designed truthful mechanism consists of an allocation rule (i.e., collaborator selection) $g(\boldsymbol{b}) = \{g_{i,j}(\boldsymbol{b})\}_{i\in\mathcal{N},\, j\in\mathcal{M}_1}$ and a pricing or reward policy $p(\boldsymbol{b}) = \{p_i(\boldsymbol{b})\}_{i\in\mathcal{N}}$, where the function $g_{i,j}(\boldsymbol{b})$ specifies winners that can offload tasks, while the function $p_i(\boldsymbol{b})$ stipulates the amount of rewards or reimbursements to task executors. Therefore, the utility of any requester i can be calculated as

$$U_i(\boldsymbol{b}) = \sum_{j\in\mathcal{M}_1} g_{i,j}(\boldsymbol{b})b_{i,j} - p_i(\boldsymbol{b}). \tag{1}$$

Note that in the bidding process, each requester may strategically misreport its bid in order to maximize its utility. As such, in the proposed truthful mechanism design, the following three properties have to be met to prevent misreport and incentivize participation.

Definition 1 (Individual Rationality (IR)): A truthful mechanism $M^R = (g(\boldsymbol{b}), p(\boldsymbol{b}))$ is individually rational for all requesters, if their utilities are no less than 0, i.e.,

$$U_i(\boldsymbol{b}) \geq 0, \quad \forall\, i \in \mathcal{N}. \tag{2}$$

The property of individual rationality guarantees that each participator can obtain non-negative benefit which provides them incentives to participate.

Definition 2 (Incentive Compatibility (IC)): A truthful mechanism $M^R = (g(\boldsymbol{b}), p(\boldsymbol{b}))$ is incentively compatible if no requester can improve its utility by misreporting its bid, i.e.,

$$U_i(b_i, \boldsymbol{b}_{-i}) \geq U_i(\hat{b}_i, \boldsymbol{b}_{-i}), \quad \hat{b}_i \in \eta(i),\ \forall\, i \in \mathcal{N}, \tag{3}$$

where $\boldsymbol{b}_{-i} \in \boldsymbol{b}$ is defined as the set of bids from other requesters except i, and $\eta(i)$ as the set of potential misreport of requester i. In the real-world scenario, this property is of great importance as it can force all bidders to bid their real bids. It resists market manipulation and ensures auction efficiency and fairness.

Definition 3 (Budget Balance (BB)): A truthful mechanism $M^R = (g(\boldsymbol{b}), p(\boldsymbol{b}))$ is budget balanced if no requesters' payments exceed their budgets, i.e.,

$$p_i(\boldsymbol{b}) \leq B_i,\ \forall\, i \in \mathcal{N}. \tag{4}$$

Besides, since each task executor has limited computation resource capacity, we impose the following constraint on each task executor as

$$\sum_{i\in\mathcal{N}} f_i g_{i,j}(\boldsymbol{b}) \leq F_j,\ \forall\, j \in \mathcal{M}_1, \tag{5}$$

where F_j is the maximal computation resource at executor j. Furthermore, by considering the fact that each mobile user can be offloaded to at most one task executor, and each collaborator can only execute at most one offloaded task, we have the following constraints:

$$\sum_{i\in\mathcal{N}} g_{i,j}(\boldsymbol{b}) \leq 1,\ \forall\, j \in \mathcal{M}, \tag{6}$$

$$\sum_{j\in\mathcal{M}_1} g_{i,j}(\boldsymbol{b}) \leq 1,\ \forall\, i \in \mathcal{N}. \tag{7}$$

Note that constraints (6) and (7) exclude the BS since the BS can accept more than one task resulting from its ample computational capacity.

B. Problem Formulation

Similar to [35], [39], in this paper, we aim to derive functions $g(\boldsymbol{b})$ and $p_i(\boldsymbol{b})$ that maximize the expected net revenue, which can be expressed as

$$\mathcal{L}(g(\boldsymbol{b}), p(\boldsymbol{b})) = \mathbb{E}_{\boldsymbol{b}\sim\mathbb{F}}\{\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}_1}(p_i(\boldsymbol{b}) - c_{i,j})g_{i,j}(\boldsymbol{b})\}, \tag{8}$$

where $\mathbb{F} = F_1 \times F_2 \times \cdots \times F_N$ is the joint distribution of all requesters' biding $\boldsymbol{b}$, $c_{i,j} = \beta_j \times \xi_j l_i f_i^2$ is the processing cost of executor j for requester i, $\beta_j,\ j \in \mathcal{M}_1$ is the cost for unit energy consumption at task executor j, and ξ_j is the energy consumption coefficient [38]. In the following discussions, we omit the vector $\boldsymbol{b}$ in both $g(\boldsymbol{b})$ and $p_i(\boldsymbol{b})$ for notational simplification. Then, our revenue maximization mechanism design can be formulated as the following optimization problem:

$$\max_{(g,p)\in\mathcal{C}} \mathcal{L}(g,p)$$
$$\text{s.t. } (2), (3), (4), (5), (6), \text{and } (7), \tag{P1}$$

where $\mathcal{C}$ denotes a class of mechanisms. However, solving such optimization problem (P1) is extremely difficult because 1) even if constraints (4)–(7) are removed from the problem (P1), distributions on requesters' valuations have to be known to solve this problem [39]. However, such distributions are not always available in practice; 2) even though such *a prior* distributions are known in advance, according to [7], existing approaches can only deal with the problem (P1) without considering the constraint (4) and in a small scale case;[3] 3) (g, p) in (P1) are not simple variables, but functions, which make (P1) fall in the scope of functional analysis. To address all aforementioned challenges, in the following section, we will propose a new mechanism design method to solve this complex problem.

IV. Truthful Deep Mechanism Design

In this section, we first reformulate (P1), and then develop a framework to design a truthful deep mechanism to maximize the net revenue with the consideration of all constraints on IR, IC, BB, and computation capacity.

A. Problem Reformulation

We first introduce the following metrics to measure the deviation degree from IR, IC, BB, and the constraint (5). For these reasons, we

- define expected ex-post regret for any requester i as

$$\rho(g, p_i) = \mathbb{E}_{\boldsymbol{b}\sim\mathbb{F}}\{\max_{\hat{b}_i\in\eta(i)} U_i(\hat{b}_i, \boldsymbol{b}_{-i}) - U_i(b_i, \boldsymbol{b}_{-i})\}, \tag{9}$$

[3]Small scale means the number of participants in the bidding campaign is small, such as two or three.

If $\rho(g, p_i) = 0$, requester i cannot gain its utility by misreporting its information, which means the constraint (3) holds.

- define the expected ex-post IR penalty to requester i as

$$\delta(g, p_i) = \mathbb{E}_{\boldsymbol{b}\sim\mathbb{F}}\{\max\{0, -U_i(\boldsymbol{b})\}\}. \tag{10}$$

If $\delta(g, p_i) = 0$, the utility of requester i is no less than zero, which means the constraint (2) satisfies.

- define the expected BB penalty to requester i as

$$\phi(g, p_i) = \mathbb{E}_{\boldsymbol{b}\sim\mathbb{F}}\{\max\{0, p_i(\boldsymbol{b}) - B_i\}\}. \tag{11}$$

If $\phi(g, p_i) = 0$, no requesters can have an overpayment, which means the constraint (4) satisfies.

- define the expected computation resource (CR) penalty of task executor j as follows

$$\theta_j(g) = \mathbb{E}_{\boldsymbol{b}\sim\mathbb{F}}\left\{\max\left\{0, \sum_{i\in\mathcal{N}} f_i g_{i,j}(\boldsymbol{b}) - F_j\right\}\right\}. \tag{12}$$

If $\theta_j(g) = 0$, no executor exceeds its computation capacity, which means the constraint (5) satisfies.

Furthermore, since the joint distribution, i.e., $\mathbb{F}$, of all requesters' biding is unknown to the BS, in order to further simplify the mechanism problem (P1), we use the geometric average to represent (9)–(12) and the objective. To this end, we apply a Q-length sample set $TS = \{\boldsymbol{b}^{(1)}, \ldots, \boldsymbol{b}^{(q)}, \ldots, \boldsymbol{b}^{(Q)}\}$ from the historical data and estimate (8)–(12) based on this set as

$$\hat{\delta}(g, p_i) = \frac{1}{Q}\sum_{q=1}^{Q} \max\{0, -U_i(\boldsymbol{b}^{(q)})\}, \tag{13}$$

$$\hat{\phi}(g, p_i) = \frac{1}{Q}\sum_{q=1}^{Q} \max\{0, p_i(\boldsymbol{b}^{(q)}) - B_i\}, \tag{14}$$

$$\hat{\theta}_j(g) = \frac{1}{Q}\sum_{q=1}^{Q} \max\left\{0, \sum_{i\in\mathcal{N}} f_i g_{i,j}(\boldsymbol{b}^{(q)}) - F_j\right\}. \tag{15}$$

Moreover, in order to estimate the constraint (9), another sample set $TS^{(q)}$ independently drawn from the historical data is leveraged as misreport bids for each $\boldsymbol{b}^{(q)}$, and we calculate the maximum utility gain over this sample set $TS^{(q)}$ as

$$\hat{\rho}(g, p_i) = \frac{1}{Q}\sum_{q=1}^{Q} \max_{\hat{b}_i \in TS^{(q)}} U_i(\hat{b}_i, \boldsymbol{b}_{-i}^{(q)}) - U_i(\boldsymbol{b}^{(q)}), \tag{16}$$

where $\boldsymbol{b}_{-i}^{(q)}$ represents the q^{th} element in the sample set $TS^{(q)}$, but not including the bids of requester i. Note that in order to keep in line with the traditional usage in machine learning, we intend to minimize the negative expected net revenue so that the estimation of (8) can be expressed as

$$\widehat{\mathcal{NL}}(g, p) = -\frac{1}{Q}\sum_{q=1}^{Q}\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}_1} (p_i(\boldsymbol{b}^{(q)}) - c_{i,j}) g_{i,j}(\boldsymbol{b}^{(q)}). \tag{17}$$

Given these estimates, we can reformulate the original problem (P1) as

$$\begin{aligned} &\min_{(g,p)\in\mathcal{C}} \widehat{\mathcal{NL}}(g, p) \\ &\text{s.t.}\ \ \hat{C}_1 : (6),\ (7), \\ &\quad\ \ \hat{C}_2 : \hat{\theta}_j(g) = 0\ \forall j \in \mathcal{M}_1, \\ &\quad\ \ \hat{C}_3 : \hat{\delta}(g, p_i) = 0,\ \forall i \in \mathcal{N}, \\ &\quad\ \ \hat{C}_4 : \hat{\rho}(g, p_i) = 0,\ \forall i \in \mathcal{N}, \\ &\quad\ \ \hat{C}_5 : \hat{\phi}(g, p_i) = 0,\ \forall i \in \mathcal{N}. \end{aligned} \tag{P2}$$

It is worth noting that g and p in the problem (P2) are optimizing functions. To this end, inspired by the framework of multi-task machine learning, we design a truthful deep mechanism in follows where g and p are both modelled by our designed inter-related neural networks, and are further trained together as a whole neural network. Furthermore, the designed truthful deep mechanism is actually a randomized mechanism, where the allocation scheme g represents the probability that a requester can be allocated to a given task executor.

B. Design of Allocation Rule and Pricing Policy

Note that in the reformulated problem (P2), our aim is to find suitable allocation rule and pricing policy, i.e., g and p, to minimize the objective while satisfying constraints $\hat{C}_1 - \hat{C}_5$. For this purpose, we model the allocation rule, i.e., g, as a feed-forward neural network which contains L fully-connected hidden layers with sigmoidal activations for each node, as shown in Fig. 2. We use the softmax activation function to output a interim vector as $a_{11}, a_{(N+1)1} \cdots z_{11}, z_{N(M+2)}$, and determine the final allocation results by comparing values of those elements in the interim vector. Specifically, in this network, we denote $y_k^{(\ell)}$ as the output of sigmoidal activation function k at the ℓ^{th} hidden layer, where $\ell \in \mathcal{L} = \{1, 2 \cdots, L\}$ and $k \in \mathcal{K}_\ell = \{1, 2, \ldots, J_\ell\}$. J_ℓ is the total number of sigmoidal activation functions at the hidden layer ℓ, and we index each hidden layer's sigmoid functions from top to bottom as $1, 2 \cdots J_\ell$. Let $\boldsymbol{w}_k^{(\ell)}$ be the row weight vector belonging to the k^{th} sigmoid function at hidden layer ℓ. We further use the row vectors $\boldsymbol{w}_{i,j,1}^{(L+1)}$ and $\boldsymbol{w}_{i,j,2}^{(L+1)}$ to represent the weights for output layer $a_{i,j}$ and $z_{i,j}$, respectively, and use $\boldsymbol{w} = \{\boldsymbol{w}_k^{(\ell)}, \boldsymbol{w}_{i,j,1}^{(L+1)}, \boldsymbol{w}_{i,j,2}^{(L+1)}\}$ to denote all vector parameters which are arranged in a row vector. Thus, the outputs of the k^{th} sigmoid function at the first and any hidden layer ℓ can be expressed as

$$d_k^{(1)} = \boldsymbol{w}_k^{(1)}\boldsymbol{I};\quad y_k^{(1)} = \sigma(d_k^{(1)}),\ \forall k \in \mathcal{K}_1, \tag{18}$$

$$d_k^{(\ell)} = \boldsymbol{w}_k^{(\ell)}\boldsymbol{y}^{(\ell-1)};\quad y_k^{(\ell)} = \sigma(d_k^{(\ell)}),\ \forall \ell \in \mathcal{L};\ k \in \mathcal{K}_\ell, \tag{19}$$

where the column vector $\boldsymbol{I} = \{I_1, I_2, \ldots, I_{N(M+3)}\}^H$ is the input of allocation rule architecture, and the sigmoid function is $\sigma(x) = \frac{1}{1+e^{-x}}$, and $\boldsymbol{y}^{(\ell)} = \{y_1^{(\ell)}, y_2^{(\ell)}, \ldots, y_{J_\ell}^{(\ell)}\}^T$ which is a column vector. The outputs of the last layer in this neural network

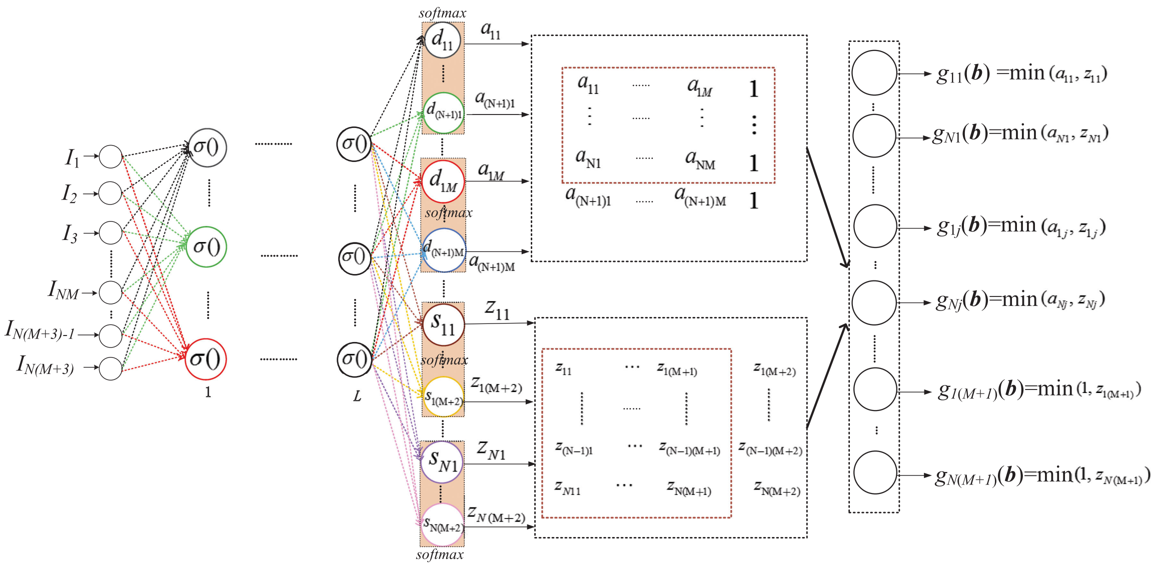
softmax
$g_{11}(\boldsymbol{b})=\min(a_{11}, z_{11})$
$g_{N1}(\boldsymbol{b})=\min(a_{N1}, z_{N1})$
$g_{1j}(\boldsymbol{b})=\min(a_{1j}, z_{1j})$
$g_{Nj}(\boldsymbol{b})=\min(a_{Nj}, z_{Nj})$
$g_{1(M+1)}(\boldsymbol{b})=\min(1, z_{1(M+1)})$
$g_{N(M+1)}(\boldsymbol{b})=\min(1, z_{N(M+1)})$

Fig. 2. Allocation rule architecture g.

are

$$d_{i,j}^{(L+1)} = \boldsymbol{w}_{i,j,1}^{(L+1)} \boldsymbol{y}^{(L)},\ \forall\, i = 1, 2 \cdots, N+1;\ \forall\, j \in \mathcal{M}, \tag{20}$$

$$\begin{aligned} s_{i,j}^{(L+1)} &= \boldsymbol{w}_{i,j,2}^{(L+1)} \boldsymbol{y}^{(L)},\ \forall\, i \in \mathcal{N};\ \forall\, j = 1, 2 \cdots, M+2, \\ a_{i,j} &= softmax(d_{1,j}^{(L+1)}, \ldots, d_{N+1,j}^{(L+1)}),\ \forall\, i \in \mathcal{N};\ \forall\, j \in \mathcal{M}_1, \\ z_{i,j} &= softmax(s_{i,1}^{(L+1)}, \ldots, s_{i,(M+2)}^{(L+1)}), \forall\, i \in \mathcal{N};\ \forall\, j \in \mathcal{M}_1, \end{aligned} \tag{21}$$

where the softmax function is defined as $softmax(x_1, \ldots, x_n) = \frac{e^{x_i}}{\sum_{k=1}^{n} e^{x_k}}$. Note that in order to indicate the cases where no requester chooses collaborator j or the BS, so that requester i fails to offload its task, $d_{N+1,j}^{(L+1)}$ and $s_{i,M+2}^{(L+1)}$ are used as the reductant inputs, respectively. Let $\mathcal{A}$ and $\mathcal{Z}$ be matrixes of all $a_{i,j}$ and $z_{i,j}$, respectively. Furthermore, by considering the fact that the BS can accept more than one task, we add one identity column vector in the last column of matrix $\mathcal{A}$. Finally, the allocation result $g_{i,j}^{\boldsymbol{w}}(\boldsymbol{b})$ can be obtained by element-wise minimization of matrix $\mathcal{A}$ and matrix $\mathcal{Z}$, i.e., $g_{i,j}^{\boldsymbol{w}}(\boldsymbol{b}) = \min\{a_{i,j}, z_{i,j}\}$.

Similarly, the pricing policy is also modeled by using a feed-forward neural network with $T = |\mathcal{T}|$ fully-connected hidden layers and a fully-connected output layer, as shown in Fig. 3. We use $\boldsymbol{w}_k^{'(t)}$ to denote the weight vector of node k at layer t in the price architecture and the output of node k at the hidden layer t is $c_k^{(t)}$. Let all outputs at the hidden layer t be a column vector, denoted as $\boldsymbol{c}^{(t)} = \{c_1^{(t)}, c_2^{(t)}, \ldots, c_{J_t'}^{(t)}\}^H$, where J_t' is the total number of sigmoidal activation functions at the hidden layer t. Also, let $\mathcal{K}_t = \{1, 2, \ldots, J_t'\}$. Likewise, the outputs of the k^{th} sigmoid function at the first layer and any hidden layer t can be respectively written as

$$h_k^{(1)} = \boldsymbol{w}_k^{'(1)} \boldsymbol{J};\quad c_k^{(1)} = \sigma(h_k^{(1)}),\ \forall\, k = 1, 2, \ldots, J_1', \tag{22}$$

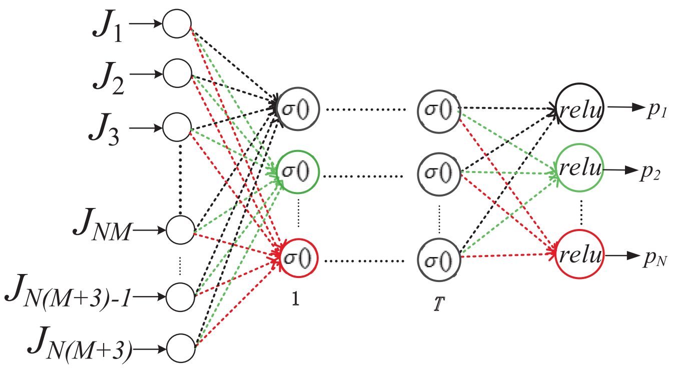

Fig. 3. Pricing policy architecture p.

$$h_k^{(t)} = \boldsymbol{w}_k^{'(t)} \boldsymbol{c}^{(t-1)};\ c_k^{(t)} = \sigma(h_k^{(t)}),\ \forall\, t \in \mathcal{T};\ \forall\, k \in \mathcal{K}_t, \tag{23}$$

where the column vector $\boldsymbol{J} = \{J_1, J_2, \ldots, J_{N(M+3)}\}^H$ is the input of pricing policy architecture. The outputs of the last layer in this neural network are

$$h_i^{(T+1)} = \boldsymbol{w}_i^{'(T+1)} \boldsymbol{c}^{(T)};\ p_i^{\boldsymbol{w}'}(\boldsymbol{b}) = relu(h_i^{(T+1)}),\ \forall\, i \in \mathcal{N}, \tag{24}$$

where $relu(x) = \max\{x, 0\}$ ensures that payments are non-negative. We use $\boldsymbol{w}' = \{\boldsymbol{w}_k^{'(t)}, t \in \mathcal{T} \cup (T+1); k \in \mathcal{K}_t\}$ to denote all vector parameters which are also arranged in a row vector, and let $p_i^{\boldsymbol{w}'}(\boldsymbol{b})$ represent the payment made by requester i given the pricing policy architecture parameters $\boldsymbol{w}'$ and the biding $\boldsymbol{b}$.

Lemma 1: According to the designed allocation rule, the allocation result, i.e., $g_{i,j}(b)$, satisfies both constraints (6) and (7).

Proof: Since the allocation rule outputs $a_{i,j}$ and $z_{i,j}$ by softmax function, as in Fig. 2, $a_{i,j}$ and $z_{i,j}$ take values between zero and one. For any collaborator j, the softmax function is calculated based on all requesters, which is the column of matrix $\mathcal{A}$. For any requester i, the softmax function is implemented based on each row of matrix $\mathcal{Z}$. Suppose that requester i has the highest probability to offload its task to collaborator j, which means $a_{i,j}$ is the largest value in the jth column in the matrix $\mathcal{A}$. Moreover, collaborator j also coincidentally has the highest chance to accept and execute task from requester i. This means $z_{i,j}$ is the largest value in the ith row in the matrix $\mathcal{Z}$. Therefore, for the other collaborator j', it is impossible to accept the task from requester i as $z_{i,j} > z_{i,j'}$. Even though $a_{i,j'}$ is still a largest one in the j' column, requester i cannot offload task to j' because of $g_{i,j'} = \min\{a_{i,j'}, z_{i,j'}\}$. The same reasons can be applied to another requester i', which cannot offload its task to the same collaborator j. Thus, the final allocation result meets both constraints (6) and (7). This completes the proof. ■

Following the similar procedures as in [40], we can easily prove the following Lemma.

Lemma 2: Let $\boldsymbol{r} = \{r_1, r_2, \ldots, r_Q\}$ be a sample drawn independently from distribution $\mathcal{D}$. Then the following inequation holds with probability of at least $1-\alpha$.

$$\mathbb{E}_{\boldsymbol{r}\in\mathcal{D}}\{f(\boldsymbol{r})\} \leq \frac{1}{Q}\sum_{i=1}^{Q} f(r_i) + 2\mathcal{R}_Q(\mathcal{F}) + \sqrt{\frac{\log\frac{1}{\alpha}}{Q}}, \quad (25)$$

where $\mathcal{R}_Q(\mathcal{F}) = \frac{1}{Q}\mathbb{E}_{\boldsymbol{\tau}}\{\sup_{f(\boldsymbol{r})\in\mathcal{F}}\sum_{r_i\in\boldsymbol{r}}\tau_i f(r_i)\}$, τ_i is a random variable, which is drawn from a uniform distribution on $\{-1, 1\}$, and set $\mathcal{F}$ is a class of functions $f(\boldsymbol{r})$.

Theorem 1: The maximal expected revenue achieved by the proposed machine learning technique is bounded by $\frac{1}{Q}\sum_{q=1}^{Q}\sum_{i\in\mathcal{N}} p_i(b^{(q)}) + 2b_{max}N\sqrt{\frac{2\log(|\mathcal{P}|)}{Q}} + \sqrt{\frac{\log\frac{1}{\alpha}}{Q}}$ with probability of at least $1-\alpha$, where $\mathcal{P}$ is the set of all reward functions.

Proof: Based on Lemma 2, we only need to calculate $\mathcal{R}_Q(\mathcal{F})$. Let $f(\boldsymbol{b}) = \sum_{i\in\mathcal{N}} p_i(\boldsymbol{b})$, and $p_i'(\boldsymbol{b})$ is another payment method from the set $\mathcal{P}'$, such that $\max_{\boldsymbol{b}}\sum_{i\in\mathcal{N}}|p_i(\boldsymbol{b}) - p_i'(\boldsymbol{b})| \leq \epsilon$. We have

$$\mathcal{R}_Q(\mathcal{P}) = \frac{1}{Q}\mathbb{E}_{\boldsymbol{\tau}}\left\{\sup_{p(\boldsymbol{b})}\sum_{q=1}^{Q}\tau_q\sum_{i\in\mathcal{N}} p_i(\boldsymbol{b}^{(q)})\right\} = \frac{1}{Q}\mathbb{E}_{\boldsymbol{\tau}}\left\{\sup_{p(\boldsymbol{b})}\sum_{q=1}^{Q}\tau_q \sum_{i\in\mathcal{N}} p_i'(\boldsymbol{b}^{(q)})\right\} + \frac{1}{Q}\mathbb{E}_{\boldsymbol{\tau}}\left\{\sup_{p(\boldsymbol{b})}\sum_{q=1}^{Q}\tau_q\sum_{i\in\mathcal{N}}|p_i(\boldsymbol{b}^{(q)}) - p_i'(\boldsymbol{b}^{(q)})|\right\}$$

$$\leq \frac{1}{Q}\mathbb{E}_{\boldsymbol{\tau}}\left\{\sup_{p(\boldsymbol{b})}\sum_{q=1}^{Q}\tau_q\sum_{i\in\mathcal{N}} p_i'(\boldsymbol{b})\right\} + \frac{\epsilon}{Q}\mathbb{E}_{\boldsymbol{\tau}}\{\sum_{q=1}^{Q}\tau_q\}$$

$$\leq \sqrt{\sum_{q=1}^{Q}\left(\sum_{i\in\mathcal{N}} p_i'(\boldsymbol{b}^{(q)})\right)^2}\frac{\sqrt{2\log(|\mathcal{P}'|)}}{Q}$$

$$\leq N b_{max}\sqrt{\frac{2\log(|\mathcal{P}'|)}{Q}},$$

where b_{max} is the maximum submitted valuation from the distribution $\mathbb{F}$, and the last inequation holds because

$$\sqrt{\sum_{q=1}^{Q}(\sum_{i\in\mathcal{N}} p_i'(\boldsymbol{b}^{(q)}))^2} \leq \sqrt{\sum_{q=1}^{Q}(Nb_{max})^2} = \sqrt{Q}Nb_{max}.$$

This completes the proof. ■

Theorem 2: The expected average ex-post regret, i.e., $\rho^A = \frac{1}{N}\sum_{i\in\mathcal{N}}\rho(g, p_i)$ achieved by the proposed machine learning technique is bounded by $\frac{1}{N}\sum_{i\in\mathcal{N}}\hat{\rho}(g, p_i) + 2b_{max}\sqrt{\frac{2\log(K(\mathcal{C},\frac{\epsilon}{2}))}{Q}} + \frac{1}{N}\sqrt{\frac{\log\frac{1}{\alpha}}{Q}}$ with probability of at least $1-\alpha$, where $K(\mathcal{C},\frac{\epsilon}{2}))$ is the minimum covering number for the truthful mechanism set $\mathcal{C}$ by a ball with radius $\frac{\epsilon}{2}$.

Proof: At beginning, we give some definitions as follows. Let $\mathcal{U}_i$ be the set of all possible utility functions for smartphone user i, i.e., $U_i(\boldsymbol{b})$, and $\mathcal{U}$ is defined as a set of all utility functions, i.e., $\{U_i(\boldsymbol{b}),\ i\in\mathcal{N}\}$. Furthermore, Define a new function $t_i(\boldsymbol{b})$ as

$$t_i(\boldsymbol{b}) = \max_{\hat{b}_i\in\eta(i)} U_i(\hat{b}_i, \boldsymbol{b}_{-i}) - U_i(b_i, \boldsymbol{b}_{-i}).$$

Let $\mathcal{T}_i$ be the set of the function $t_i(\boldsymbol{b})$ with different $U_i(b)\in\mathcal{U}_i$, and $\mathcal{T} = \{\mathcal{T}_i,\ i\in\mathcal{N}\}$. Moreover, we define a spatial distance between two different utility function $U_i(\boldsymbol{b})$ and $U_i'(\boldsymbol{b})$ as $\max_{\boldsymbol{b},\boldsymbol{b}'}|U_i(\boldsymbol{b}) - U_i'(\boldsymbol{b})|$, and a distance between $\mathcal{U}$ and $\mathcal{U}'$ as $\max_{\boldsymbol{b},\boldsymbol{b}'}\sum_{i\in\mathcal{N}}|U_i(\boldsymbol{b}) - U_i'(\boldsymbol{b})|$. We then define $K(U_i(\boldsymbol{b}),\epsilon)$, $K(\mathcal{U},\epsilon)$, $K(T_i(\boldsymbol{b}),\epsilon)$, and $K(\mathcal{T},\epsilon)$ be the minimum number of balls with radius ϵ to cover the set $U_i(\boldsymbol{b})$, $\mathcal{U}$, $T_i(\boldsymbol{b})$, and $\mathcal{T}$ under the correspondingly defined distances, respectively. In addition, the spatial distance between two mechanisms $(g_{i,j}(\boldsymbol{b}),\ p_i(\boldsymbol{b})), (g_{i,j}'(\boldsymbol{b}),\ p_i'(\boldsymbol{b}))\in\mathcal{C}$ is defined as $\max_{\boldsymbol{b}}\sum_{i\in\mathcal{N},\ j\in\mathcal{M}_1}|g_{i,j}(\boldsymbol{b}) - g_{i,j}'(\boldsymbol{b})| + \sum_{i\in\mathcal{N}}|p_i(\boldsymbol{b}) - p_i'(\boldsymbol{b})|$, and $K(\mathcal{C},\epsilon)$ is the minimum number of balls with radius ϵ to cover the mechanism set $\mathcal{C}$ under this distance. Let $t(\boldsymbol{b}) = \sum_{i\in\mathcal{N}} t_i(\boldsymbol{b})$, and the set $\overline{\mathcal{T}} = \{t(\boldsymbol{b}),\ \forall\,(t_1(\boldsymbol{b}), t_2(\boldsymbol{b}),\ldots,t_N(\boldsymbol{b}))\in\mathcal{T}\}$. Similar to Theorem 1, we assume that there is another $t'(\boldsymbol{b})\in\overline{\mathcal{T}}'$ where $|\overline{\mathcal{T}}'| \leq K(\overline{\mathcal{T}},\epsilon)$, which means $\max_{\boldsymbol{b}}|t(\boldsymbol{b}) - t'(\boldsymbol{b})| \leq \epsilon$ holds. Then, we have

$$\begin{aligned}\mathcal{R}_Q\left(\overline{\mathcal{T}}\right) &= \frac{1}{Q}\mathbb{E}_{\boldsymbol{\tau}}\left\{\sup_{t_i(\boldsymbol{b})}\sum_{q=1}^{Q}\tau_q\sum_{i\in\mathcal{N}} t_i(\boldsymbol{b}^{(q)})\right\}\\ &= \frac{1}{Q}\mathbb{E}_{\boldsymbol{\tau}}\left\{\sup_{t_i(\boldsymbol{b})}\sum_{q=1}^{Q}\tau_q\sum_{i\in\mathcal{N}} t'(\boldsymbol{b}^{(q)})\right\}\\ &\quad + \frac{1}{Q}\mathbb{E}_{\boldsymbol{\tau}}\left\{\sup_{t(\boldsymbol{b})}\sum_{q=1}^{Q}\tau_q\sum_{i\in\mathcal{N}}|t_i(\boldsymbol{b}^{(q)}) - t_i'(\boldsymbol{b}^{(q)})|\right\}\\ &\leq Nb_{max}\sqrt{\frac{2\log(K(\overline{\mathcal{T}},\epsilon))}{Q}}.\end{aligned} \quad (26)$$

From (26), we only need to prove $K(\overline{\mathcal{T}},\epsilon) \leq K(\mathcal{C},\frac{\epsilon}{2})$. In order to prove this inequation, we carry out the following three steps.

- We suppose that there exists an utility function set $\mathcal{U}_i'$ with covering number at most $K(\mathcal{U}_i,\frac{\epsilon}{2})$, such that $\max_{\boldsymbol{b}}|U_i(\boldsymbol{b}) - U_i'(\boldsymbol{b})| \leq \frac{\epsilon}{2}$. Then, for any $\boldsymbol{b}$, we have

$$|\max_{\overline{b}_i}(U_i(\overline{b}_i, \boldsymbol{b}_{-i}) - U_i(b_i, \boldsymbol{b}_{-i}))-$$

$$\max_{b_i'}(U_i'(b_i',\boldsymbol{b}_{-i})-U_i'(b_i,\boldsymbol{b}_{-i}))|$$
$$\leq |\max_{\bar{b}_i}(U_i(\bar{b}_i,\boldsymbol{b}_{-i})-\max_{b_i'}(U_i'(b_i',\boldsymbol{b}_{-i})+ U_i'(b_i,\boldsymbol{b}_{-i}))-U_i(b_i,\boldsymbol{b}_{-i}))|$$
$$\leq |\max_{\bar{b}_i}(U_i(\bar{b}_i,\boldsymbol{b}_{-i})-\max_{b_i'}(U_i'(b_i',\boldsymbol{b}_{-i})|+ |U_i'(b_i,\boldsymbol{b}_{-i}))-U_i(b_i,\boldsymbol{b}_{-i}))|$$
$$\leq |\max_{\bar{b}_i}(U_i(\bar{b}_i,\boldsymbol{b}_{-i})-\max_{b_i'}(U_i'(b_i',\boldsymbol{b}_{-i})|+\frac{\epsilon}{2}.$$

Furthermore, let $\bar{b}_i^* = \arg\max_{\bar{b}_i} U_i(\bar{b}_i,\boldsymbol{b}_{-i})$ and $b_i'^* = \arg\max_{b_i'} U_i(b_i',\boldsymbol{b}_{-i})$. Then, we have

$$\max_{\bar{b}_i} U_i(\bar{b}_i,\boldsymbol{b}_{-i}) = U_i(\bar{b}_i^*,\boldsymbol{b}_{-i}) \leq U_i'(\bar{b}_i^*,\boldsymbol{b}_{-i})+\frac{\epsilon}{2} \leq U_i'(b_i'^*,\boldsymbol{b}_{-i})+\frac{\epsilon}{2} = \max_{b_i'} U_i(b_i',\boldsymbol{b}_{-i})+\frac{\epsilon}{2},$$

$$\max_{b_i'} U_i'(b_i',\boldsymbol{b}_{-i}) = U_i'(b_i'^*,\boldsymbol{b}_{-i}) \leq U_i(b_i'^*,\boldsymbol{b}_{-i})+\frac{\epsilon}{2} \leq U_i(b_i^*,\boldsymbol{b}_{-i})+\frac{\epsilon}{2} = \max_{b_i} U_i(b_i,\boldsymbol{b}_{-i})+\frac{\epsilon}{2}.$$

Therefore, for any $U_i(\boldsymbol{b})$, there always exists $U_i'(\boldsymbol{b})$, which satisfies $|\max_{\bar{b}_i}(U_i(\bar{b}_i,\boldsymbol{b}_{-i})-U_i(b_i,\boldsymbol{b}_{-i}))-\max_{b_i'}(U_i'(b_i',\boldsymbol{b}_{-i})-U_i'(b_i,\boldsymbol{b}_{-i}))|\leq\epsilon$. This means $K(\mathcal{T}_i,\epsilon)\leq K(\mathcal{U}_i,\frac{\epsilon}{2})$ holds.

- Assume that there exists another mechanism set $\hat{\mathcal{C}}$, satisfying $|\hat{\mathcal{C}}|\leq K(\mathcal{C},\epsilon)$. Therefore, for any $(\hat{g}_{i,j}(\boldsymbol{b}),\hat{p}_i(\boldsymbol{b}))\in\hat{\mathcal{C}}$, we have
$$\max_{\boldsymbol{b}}\sum_{i,j}|g_{i,j}(\boldsymbol{b})-\hat{g}_{i,j}(\boldsymbol{b})|+\sum_i|p_i(\boldsymbol{b})-\hat{p}_i(\boldsymbol{b})|\leq\epsilon.$$
Remind that $v_{i,j}$ is the truthful valuation to smartphone user i, and for the utility functions $U_i(\boldsymbol{b})$ and $\hat{U}_i(\boldsymbol{b})$, we have
$$|U_i(\boldsymbol{b})-\hat{U}_i(\boldsymbol{b})|\leq$$
$$|\sum_{i,j}v_{i,j}g_{i,j}(\boldsymbol{b})-\sum_{i,j}v_{i,j}\hat{g}_{i,j}(\boldsymbol{b})|+|p_i(\boldsymbol{b})-\hat{p}_i(\boldsymbol{b})|\leq$$
$$||v_i||_\infty\sum_j|g_{i,j}(\boldsymbol{b})-\hat{g}_{i,j}(\boldsymbol{b})|+|p_i(\boldsymbol{b})-\hat{p}_i(\boldsymbol{b})|\leq$$
$$\sum_j|g_{i,j}(\boldsymbol{b})-\hat{g}_{i,j}(\boldsymbol{b})|+|p_i(\boldsymbol{b})-\hat{p}_i(\boldsymbol{b})|,$$
where $v_i=\{v_{i,1},v_{i,2},\ldots,v_{i,M+1}\}$, and $||v_i||_\infty$ is the infinite norm of v_i. For all smartphone users, we have
$$\sum_i|U_i(\boldsymbol{b})-\hat{U}_i(\boldsymbol{b})|\leq$$
$$\sum_{i,j}|g_{i,j}(\boldsymbol{b})-\hat{g}_{i,j}(\boldsymbol{b})|+\sum_i|p_i(\boldsymbol{b})-\hat{p}_i(\boldsymbol{b})|\leq\epsilon.$$
This means we have $K(\mathcal{U},\epsilon)\leq K(\mathcal{C},\epsilon)$.

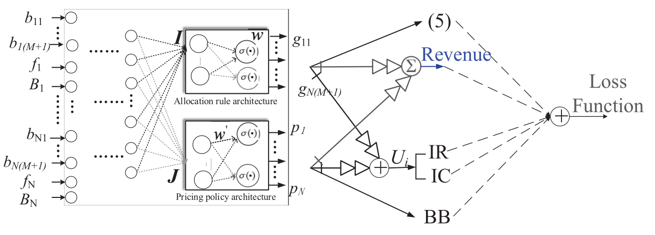

Fig. 4. The whole multi-task machine learning model of our proposed method.

- Based on the definition of $K(\mathcal{U},\epsilon)$, there exists $\hat{\mathcal{U}}$ whose covering number is at most $K(\mathcal{U},\epsilon)$. Therefore, for any $U_i(\boldsymbol{b})\in\mathcal{U}$, we have $\sum_i|U_i(\boldsymbol{b})-\hat{U}_i(\boldsymbol{b})|\leq\epsilon$. Furthermore, we have $|\sum_i U_i(\boldsymbol{b})-\sum_i\hat{U}_i(\boldsymbol{b})|\leq\epsilon$, which means that $K(\overline{\mathcal{T}},\epsilon)\leq K(\mathcal{T},\epsilon)$. Moreover, following previous two steps, we have $K(\overline{\mathcal{T}},\epsilon)\leq K(\mathcal{U},\frac{\epsilon}{2})\leq K(\mathcal{C},\frac{\epsilon}{2})$.

By applying Lemma 2, and taking into consideration of $t(\boldsymbol{b})$, we have

$$\rho^A\leq\frac{1}{N}\sum_{i\in\mathcal{N}}\hat{\rho}(g,p_i)+2b_{max}\sqrt{\frac{2\log(K(\mathcal{C},\frac{\epsilon}{2}))}{Q}}+\frac{1}{N}\sqrt{\frac{\log\frac{1}{\alpha}}{Q}}.$$

This completes the proof. ■

Note that from Theorems 1 and 2, when the number of samples, i.e., Q, is large enough, the expected revenue and the expected average ex-post regret equal the empirical revenue and the empirical average regret.

C. The Design of Training Method

After we have designed the architectures for both the allocation rule and pricing policy, we can integrate them into a single one, as shown in Fig. 4. Since the allocation rule and pricing policy architectures respectively output allocation rule, i.e., $g(\boldsymbol{b})$, and pricing policy, i.e., $p(\boldsymbol{b})$, the utility of each requester can be obtained based on (1). Then, IR, IC, BB, and the constraint (5) can be formulated as constraints in the problem (P1). Note that Fig. 4 specifies a multi-task machine learning model resulting from the facts that 1) the determinations of the allocation and pricing rules are two related tasks; 2) the first few layers are shared by two architectures, and these two architectures should be trained simultaneously to maximize the net revenue under our defined loss function below. According to Fig. 4 and the problem (P2), we can formulate a new optimization problem (P3) in which the optimization variables are the parameters involved in both allocation rule and pricing policy architectures, i.e., $\boldsymbol{w}$ and $\boldsymbol{w}'$, as follows:

$$\min_{\boldsymbol{w},\boldsymbol{w}'}\widehat{\mathcal{NL}}(g^{\boldsymbol{w}},p^{\boldsymbol{w}'}) \tag{P3}$$
$$\text{s.t.}\quad \hat{\rho}(g^{\boldsymbol{w}},p_i^{\boldsymbol{w}'})=0,\forall i\in\mathcal{N}, \tag{27}$$
$$\hat{\delta}(g^{\boldsymbol{w}},p_i^{\boldsymbol{w}'})=0,\ \forall i\in\mathcal{N}, \tag{28}$$
$$\hat{\phi}(g^{\boldsymbol{w}},p_i^{\boldsymbol{w}'})=0,\ \forall i\in\mathcal{N}, \tag{29}$$
$$\hat{\theta}_j(g^{\boldsymbol{w}})=0,\ \forall j\in\mathcal{M}_1. \tag{30}$$

We solve this training problem (P3) by using the augmented Lagrangian method [41]. Specifically, we first construct a Lagrangian function, i.e., loss function, and then add four quadratic penalty terms in order not to violate constraints (27), (28), (29), and (30). Thus, we have

$$Lag(\boldsymbol{w}, \boldsymbol{w}', \boldsymbol{\lambda}, \kappa) = \widehat{\mathcal{NL}}(g^{\boldsymbol{w}}, p^{\boldsymbol{w}'}) + \sum_{i\in\mathcal{N}}(\lambda_{1,i}\hat{\delta}(g^{\boldsymbol{w}}, p_i^{\boldsymbol{w}'}) + \lambda_{2,i}\hat{\phi}(g^{\boldsymbol{w}}, p_i^{\boldsymbol{w}'}) + \lambda_{3,i}\hat{\rho}(g^{\boldsymbol{w}}, p_i^{\boldsymbol{w}'})) + \sum_{j\in\mathcal{M}_1}\lambda_j\hat{\theta}_j(g^{\boldsymbol{w}}) + \frac{\kappa}{2}\left(\sum_{i\in\mathcal{N}}(\hat{\delta}(g^{\boldsymbol{w}}, p_i^{\boldsymbol{w}'})^2 + \hat{\phi}(g^{\boldsymbol{w}}, p_i^{\boldsymbol{w}'})^2 + \hat{\rho}(g^{\boldsymbol{w}}, p_i^{\boldsymbol{w}'})^2) + \sum_{j\in\mathcal{M}_1}\hat{\theta}_j(g^{\boldsymbol{w}})^2\right),$$

where $\boldsymbol{\lambda} = \{\lambda_{1,1}, \ldots, \lambda_{1,N}, \lambda_{2,1}, \ldots, \lambda_{2,N}, \lambda_{3,1}, \ldots, \lambda_{3,N}, \lambda_1, \ldots, \lambda_{M+1}\}$ are the Lagrange multipliers associated with constraints (27), (28), (29), and (30), and $\kappa > 0$ is the penalty parameter that controls the weight for violating constraints (27), (28), (29), and (30). We then perform the following updates in each iteration s:

$$(\boldsymbol{w}^{(s+1)}, \boldsymbol{w}^{'(s+1)}) = \arg\min_{\boldsymbol{w},\boldsymbol{w}'} Lag(\boldsymbol{w}, \boldsymbol{w}', \boldsymbol{\lambda}, \kappa) \quad (31)$$

$$\begin{aligned}\lambda_{1,i}^{(s+1)} &= \lambda_{1,i}^{(s)} + \kappa\hat{\delta}(g^{\boldsymbol{w}}, p_i^{\boldsymbol{w}'}),\\ \lambda_{2,i}^{(s+1)} &= \lambda_{2,i}^{(s)} + \kappa\hat{\phi}(g^{\boldsymbol{w}}, p_i^{\boldsymbol{w}'}),\\ \lambda_{3,i}^{(s+1)} &= \lambda_{3,i}^{(s)} + \kappa\hat{\rho}(g^{\boldsymbol{w}}, p_i^{\boldsymbol{w}'}),\\ \lambda_j^{(s+1)} &= \lambda_j^{(s)} + \kappa\hat{\theta}_j(g^{\boldsymbol{w}}),\end{aligned} \quad (32)$$

where the mini-batch stochastic subgradient descent is applied to solve the problem (31). For clarity, the training algorithm for allocation and pricing architectures is listed in Algorithm 1.

Next, we evaluate the computational complexity of the training method by using the same metric as in [42], [43]. We use one of neural networks as an example, which has N_I and N_o numbers of input and output nodes, respectively, and D number of hidden layers with N_ℓ nodes for each layer. Then, the computation cost for mini-batch gradient descent algorithm is $O(N_I N_\ell + N_\ell^2 D + N_\ell N_o)$. In our case, the total computation cost is $O((3NMN_\ell + 3NN_\ell + N_\ell M + N_\ell^2(L+T))N_B \times N_\lambda \times \lfloor \frac{N_D}{N_B} \rfloor)$, where the symbol $\lfloor x \rfloor$ denotes the flooring of real number x, N_λ is the maximum iteration number of λ, and N_D and N_B are the total training data and min-batch sizes, respectively. As the algorithm needs the memory space to store the gradients of the weights in the back propagation process [44], the total memory cost is $O((L+T) \times N_\ell)$. Moreover, since the BS has strong computation capacity in the mobile edge computing system, the training algorithm can be done at the BS [45].

Algorithm 1: Training Algorithm.

```
1  Initialization;
2  $w^{(0)} = w_0, w'^{(0)} = w_0'$;
3  $\lambda_{1,i}^{(0)} = \lambda_1, \lambda_{2,i}^{(0)} = \lambda_2, \lambda_{3,i}^{(0)} = \lambda_3, \forall i \in \mathcal{N}, \lambda_j^{(0)} = \lambda_4, \forall j \in \mathcal{M}_1$;
4  $\kappa = \kappa_0, n = 0$;
5  while $n \le N_\kappa$ do
6      $s = 0$;
7      while $s \le N_\lambda$ do
8          $k = 1$ ;
9          while $k \le N_k$ do
10             Obtain $w^{(s+1)}$ and $w'^{(s+1)}$ by optimizing (31) with $N_k$ numbers of mini-batch;
11         Calculate $\hat{\delta}(g^w, p_i^{w'})$, $\hat{\phi}(g^w, p_i^{w'})$, $\hat{\rho}(g^w, p_i^{w'})$, and $\hat{\theta}_j(g^w)$ by using all test data, and update $\lambda_{1,i}^{(s)}, \lambda_{2,i}^{(s)}, \lambda_{3,i}^{(s)}, \lambda_j^{(s)}$ by (32) ;
12     Update $\kappa$ with $N_\kappa$ numbers totally by $\kappa = \kappa + \frac{0.5}{n}$
13 Output;
14 $w$, $w'$, and $\hat{\mathcal{NL}}$ ;
```

V. Numerical Results

In this section, we present simulation results[4] to demonstrate the effectiveness of our proposed mechanism in EC collaborative offloading. We use Tensorflow for implementing the deep learning algorithm and assume that the number of requesters $N = 5$ and the number of collaborators $M = 4$. In our simulation, two architectures share one common layer, and both allocation rule and pricing policy architectures include 2 hidden layers with 8 nodes each. The batch size in the mini-batch stochastic gradient algorithm is set to 32 for training the allocation rule and pricing policy architectures. In each epoch, we first train the proposed mechanism on the training set and evaluate the revenue on the test set, both of which contain 64000 training samples. In addition, the architectures will keep training with the training data for ten epochs until it converges. The processes of updating $\lambda_{1,i}$, $\lambda_{2,i}$, $\lambda_{3,i}$, and λ_j run every 50 iterations with the update of the architectures. κ is updated as $\kappa = \kappa + \frac{0.5}{n}$, where n represents the number of times for updating κ, and κ is updated every 1000 iterations of updating networks. The initial value of κ is 0.5 for both DU I and CU distributions, which will be introduced later, while for DU II distribution, the initial value of κ is set to 1.0 for the purpose of fast convergence. The energy consumption coefficient, i.e., ξ_j, at task executor j is set to 10^{-26} [2], the size of offloading tasks, i.e., l_i, are randomly generated between 10 to 30 MB, and the unit energy cost, i.e., $\beta_j = 0.1$ [2]. Besides, the maximal computation capacities at collaborators and the BS are 2 GHz and 10 GHz, respectively, and the requested computation frequency is randomly produced between [0.4 GHz, 1.44 GHz] [24]. Moreover, define $\overline{\rho}$, $\overline{\delta}$, and $\overline{\phi}$ as the average values of $\hat{\rho}_i$, $\hat{\delta}_i$, and $\hat{\phi}_i$ across all requesters on

[4]Since no practical data are available, similar to [5], [6], we generate the training and test data from the three uniform distributions (i.e., two discrete uniform distributions and one continuous distribution).

the test set. Note that $\overline{\rho}$, $\overline{\phi}$, $\overline{\delta}$ and $\overline{\theta}$ represent the indicators of IC, IR, BB, and the constraint (5), respectively, which can be mathematically expressed as.

$$\overline{\rho} = \frac{1}{N}\sum_{i=1}^{N}\hat{\rho}(g,p_i), \quad \overline{\delta} = \frac{1}{N}\sum_{i=1}^{N}\hat{\delta}(g,p_i),$$

$$\overline{\phi} = \frac{1}{N}\sum_{i=1}^{N}\hat{\phi}(g,p_i), \quad \overline{\theta} = \frac{1}{M+1}\sum_{j=1}^{M+1}\hat{\theta}_j.$$

The values of Lagrangian function, revenue, $\overline{\rho}$, $\overline{\phi}$, $\overline{\delta}$ and $\overline{\theta}$ are recorded every batch in the mini-batch stochastic gradient during the training processes. In order to avoid the issue of overfitting in the training process, all results in the following figures are output on the test set after each training epoch. In addition, to study the effect of requesters' different budgets on the revenue, we generate budgets based on two different forms where the first form is a uniform distribution that contains the maximal valuation of each requester, while the other one is also a uniform distribution that any values from this distribution is higher than the maximal valuation of each requester. Furthermore, for better illustrating the generalization of our designed mechanism, we generate requesters' valuations from the following distributions. Note that our proposed mechanism has no limitation on certain distributions.

- Discrete Uniform I (DU I): valuations of each requester are drawn from the identical uniform distribution over two values $(0.5, 1.0)$.
  - Budget I (B I): the budgets of each requester are drawn from an identical uniform distribution over $[0.5, 3]$;
  - Budget II (B II): the budgets of each requester are drawn from an identical uniform distribution over $[1, 3]$;

  Discrete Uniform II (DU II): valuations are drawn from the identical uniform distribution over three values $(0.5, 1.0, 1.5)$.
  - Budget I (B I): the budgets of each requester are drawn from the identical uniform distribution over $[0.5, 5]$;
  - Budget II (B II): the budgets of each requester are drawn from the identical uniform distribution over $[1.5, 3]$;

  Continuous Uniform (CU): valuations of each requester are drawn from the identical uniform distribution over $[0, 1]$.
  - Budget I (B I): the budgets of each requester are drawn from the identical uniform distribution over $[0, 3]$;
  - Budget II (B II): the budgets of each requester are drawn from the identical uniform distribution over $[1, 3]$;

Fig. 5 illustrates the trend of Lagrangian functions through the training processes for different distributions with two budget cases. We can see that the curves of these three distributions decrease, and then gradually stay unchanged after one thousand iterations, which indicates that the networks have been trained to convergence.

Fig. 6 evaluates the revenues along the training process. As shown in this figure, all curves reach saturation after a short period of time. Moreover, among a certain budget distribution, DU II achieves the largest revenue. It is because the bidding valuations and the budgets for DU II are larger than other scenarios so that the winning probability of requesters with high valuations increases. Similarly, the revenue of CU with B I is the smallest because both bidding valuations and budgets are the smallest in average so that the reimbursements to collaborators or the BS are relatively low. Interestingly, even if the lower bound of budget distribution is higher than the upper bound of valuation distributions, i.e., using B II, the revenue of DU I is smaller than that of its counterpart using in B I. It is because by using B II, only two requesters can win and offload their tasks to nearby collaborators, as shown in Table II, so that the total revenue of DU I becomes lowest. Furthermore, for both DU II and CU, when adopting budget distribution B II, the revenue values will be higher than that of its counterpart using B I. This is because requesters with larger budgets are more easier to win and make higher payments in this case.

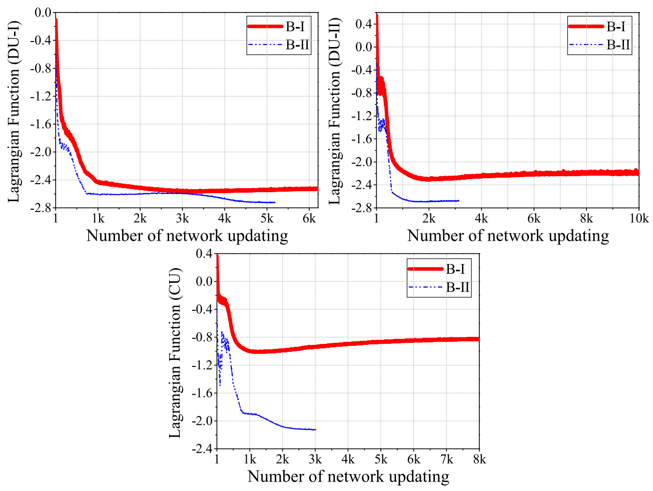

Fig. 5. The value of Lagrangian function of 3 valuation distributions with two different budgets during network updatings.

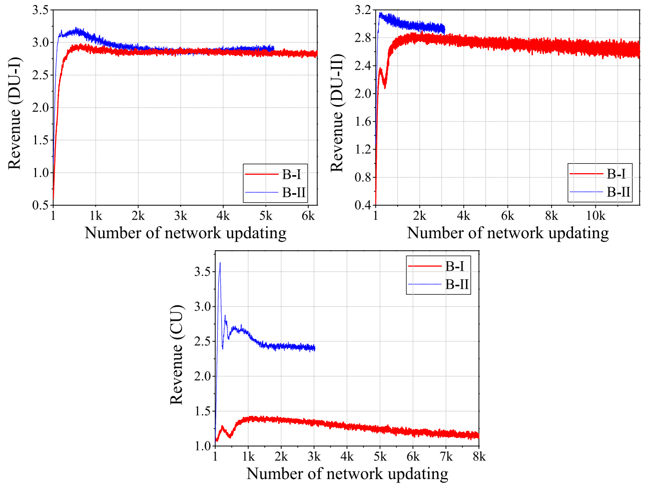

Fig. 6. The value of Revenue of 3 valuation distributions with two different budgets during network updating.

Fig. 7 depicts the trend of average Regret, i.e., $\overline{\rho}$, for the three valuation distributions with different budgets. In this picture, $\overline{\rho}$ are initially large for all cases. This may because requesters would like to get more utilities by misreporting their bidding information. However, for all three valuation distributions, the

TABLE I
THE VALUES OF WELL-TRAINED AVERAGE REGRET AND IR PENALTY

	DU I		DU II		CU	
	Regret ($\overline{\rho}$)	IR penalty ($\overline{\delta}$)	Regret ($\overline{\rho}$)	IR penalty ($\overline{\delta}$)	Regret ($\overline{\rho}$)	IR penalty ($\overline{\delta}$)
B I	0.0047	0.0045	0.0041	0.0035	0.0038	0.0025
B II	0.0042	0.0044	0.0039	0.0038	0.0040	0.0030

TABLE II
OFFLOADING ALLOCATION RESULTS BY PROPOSED INCENTIVE MECHANISM FOR THREE VALUATION DISTRIBUTIONS WITH TWO DIFFERENT BUDGETS

			DU I					DU II					CU				
			Collaborator				BS	Collaborator				BS	Collaborator				BS
			1	2	3	4		1	2	3	4		1	2	3	4	
B I	Requester	1			√												√
		2							√					√			
		3		√						√						√	
		4	√														
		5									√						
B II	Requester	1						√									√
		2				√											
		3											√				
		4			√				√						√		
		5										√					√
√ indicates that the requester will offload its task to the corresponding BS or collaborator.																	

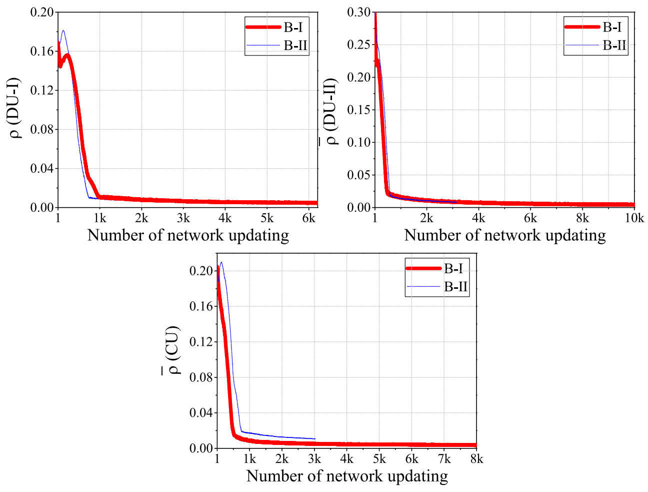

Fig. 7. The value of $\overline{\rho}$ of 3 valuation distributions with two different budgets during network updatings.

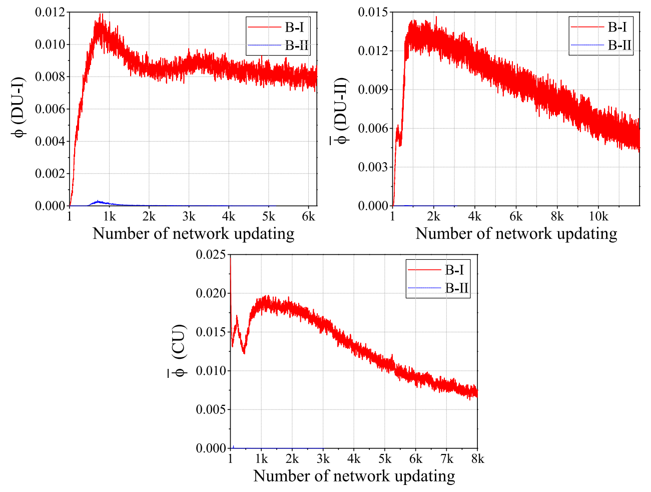

Fig. 8. The value of $\overline{\phi}$ of 3 valuation distributions with two different budgets during network updatings.

value of $\overline{\rho}$ decreases rapidly and then converges to almost zero, which means the IC condition is satisfied.

Fig. 8 shows the value of BB penalty, i.e., $\overline{\phi}$, with the increase of training process for three valuation distributions with different budgets. It can be shown that initiatively, when the budget distributions which contain the valuation distributions are considered, i.e., using B I, $\overline{\phi}$ are small enough for both DU I and DU II, while the value of $\overline{\phi}$ for CU is a bit higher than others. This is because, at the very beginning, $p^{\omega'}$ in the pricing policy architecture for both DU I and DU II is very small, but is relatively larger for CU, which can be verified by initial revenues in Fig. 6. Afterwards, $\overline{\phi}$ for those three distributions will eventually reach a plateau before a peak appears at about one thousand iterations. These can be explained as follows. As the ongoing of the training process, all revenues firstly increase so that payments from requesters may violate their budgets. Furthermore, although revenues become stable after about 1000 iterations, the training algorithm is still running to satisfy the BB constraint. As a consequence, a peak of $\overline{\phi}$ will appear shortly for all distributions, and then the values go down and stabilize below 0.01 afterwards, which means the BB condition is satisfied. Note that quite different from the trends by using B I for three valuation distributions, values of $\overline{\phi}$ are all around zero in B II case. This is because the lowest value from budget B II is higher than that of the upper bound of those valuation distributions so that the BB condition is naturally demanded in this case.

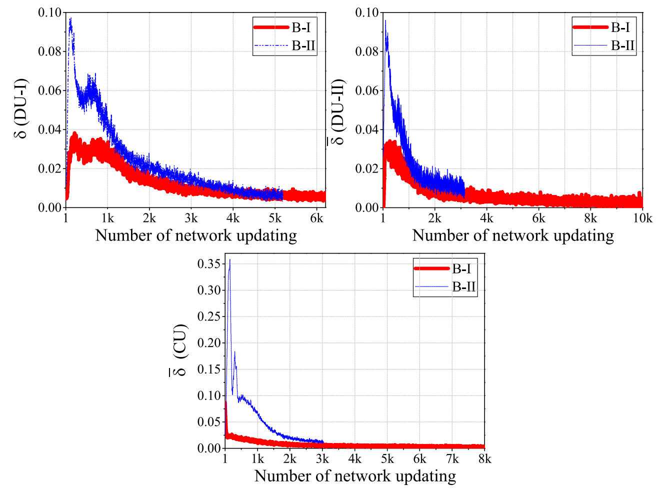

Fig. 9. The value of $\bar{\delta}$ of 3 valuation distributions with two different budgets during network updatings.

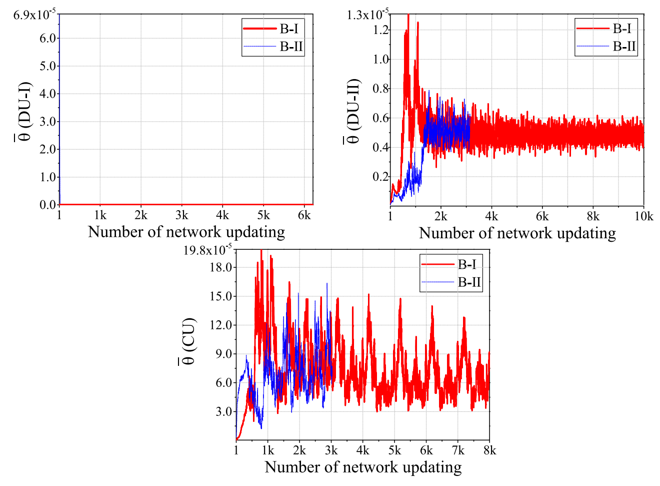

Fig. 10. The value of $\bar{\theta}$ of 3 valuation distributions with two different budgets during network updatings.

Fig. 9 demonstrates the change of IR penalty, i.e., $\bar{\delta}$, for three valuation distributions with different budgets. From this figure, we can see that with the increase of iterations, the values of $\bar{\delta}$ for all cases gradually decline, and then tend to almost zero, which means the IR condition is satisfied after training. Note that for all valuation distributions, different budget distributions have no influences on satisfying IR condition. Furthermore, Fig. 10 shows the changing trend of CR penalty, i.e., $\bar{\theta}$, for three valuation distributions. It can be seen that allocated computation resources incurs a very small violations on θ_j, which means the constraint (5) holds.

For more intuitive understanding the convergence of our trained networks, we tabulate the well-trained values of IC and IR, i.e., regret and IR penalty, in Table I. As we can see, the values of both the regret and IR penalty are small enough regardless of the selected distributions of budget balance. Moreover, Table II lists one of the potential allocation results for three valuation distributions with two budget distributions by using the well trained allocation rule and pricing policy architectures. It is worth noting that in order to get the maximum revenue, requesters may prefer more to offload their tasks to nearby collaborators rather than offload to the BS. This means offloading all tasks to the BS is not always the best choice for requesters, which further shows potential applications of the collaborative offloading architecture.

VI. Conclusion

In this paper, we design a truthful deep mechanism for cooperative task offloading in the edge computing system. Our objective is to maximize the net revenue of the service provider while satisfying IR, IC, BB, and computation resource capacity conditions. The designed mechanism is extremely difficult and no existed methods can be applied to address it effectively. To this end, inspired by multi-task machine learning model, we apply the deep learning technology to achieve our design. Specifically, two specified neural networks architectures are designed to figure out the functions of allocation rule and pricing policy, respectively, and then these neural networks are trained together by the augmented Lagrangian method. Numerical results demonstrate the effectiveness of our designed truthful deep mechanism while only incurring small IR penalty, Regret, BB penalty, and CR penalty.

References

[1] X. Chen, L. Pu, L. Gao, W. Wu, and D. Wu, "Exploiting massive D2D collaboration for energy-efficient mobile edge computing," *IEEE Wireless Commun.*, vol. 24, no. 4, pp. 64–71, Aug. 2017.

[2] G. Li and J. Cai, "An online incentive mechanism for collaborative task offloading in mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 624–636, Jan. 2020.

[3] D. Llias *et al.*, "Efficiency-revenue trade-offs in auctions," in *Int. Colloq. Automata, Lang., Program.*, 2012, pp. 488–499.

[4] A. Kiani and N. Ansari, "Towards hierarchical mobile edge computing: An auction-based profit maximization approach," *IEEE Internet Things*, vol. 4, no. 6, pp. 2082–2091, Dec. 2017, doi: 10.1109/JIOT.2017.2750030.

[5] H. Zhang, F. Guo, H. Ji, and C. Zhu, "Combinational auction-based service provider selection in mobile edge computing networks," *IEEE Access*, vol. 5, pp. 13455–13464, 2017.

[6] A. L. Jin, W. Song, and W. H. Zhuang, "Auction-based resource allocation for sharing cloudlets in mobile cloud computing," *IEEE Trans. Emerg. Top. Comput.*, vol. 6, no. 1, pp. 45–57, Jan.–Mar. 2018.

[7] T. Sandholm and A. Likhodedov, "Automated design of revenue-maximizing combinatorial auctions," *Oper. Res.*, vol. 63, no. 5, pp. 1000–1025, Oct. 2015.

[8] Y. Cai, C. Daskalakis, and S. Matthew Weinberg, "An algorithmic characterization of multi-dimensional mechanisms," in *Proc. 44th ACM Symp. Theory Comput.*, 2012, pp. 459–478.

[9] S. Bhattacharya, G. Goel, S. Gollapudi, and K. Munagala, "Budget-constrained auctions with heterogeneous items," *Theory Comput.*, vol. 8, no. 20, pp. 429–460, 2012.

[10] Y. Cai, C. Daskalakis, and S. Matthew Weinberg, "Optimal multi-dimensional mechanism design: Reducing revenue to welfare maximization," in *Proc. 53rd IEEE Symp. Found. Comput. Sci.*, 2012, pp. 130–139.

[11] C. Daskalakis, N. R. Devanur, and S. Matthew Weinberg, "Revenue maximization and ex-post budget constraints," *ACM Trans. Econ. Comput.*, vol. 6, no. 3/4, pp. 1–19, 2018.

[12] Y. Cai, C. Daskalakis, and S. Matthew Weinberg, "Understanding incentives: Mechanism design becomes algorithm design," in *Proc. 54th IEEE Symp. Found. Comput. Sci.*, 2013, pp. 618–627.

[13] A. Yao, "On solutions for the maximum revenue multi-item auction under dominant strategy and Bayesian implementations," Cornell Univ., pp. 1–28, 2016, *arXiv abs/1607.03685*.

[14] M. Al-Ayyoub and H. Gupta, "Truthful spectrum auctions with approximate social-welfare or revenue," *IEEE Trans. Netw.*, vol. 22, no. 6, pp. 1873–1885, Dec. 2014.

[15] C. Avery and T. Hendershott, "Bundling and optimal auctions of multiple products," *Rev. Econom. Stud.*, vol. 67, pp. 483–497, 2000.

[16] Z. Zheng, F. Wu, X. Gao, H. Zhu, S. Tang, and G. Chen, "A budget feasible incentive mechanism for weighted coverage maximization in

mobile crowdsensing," *IEEE Trans. Mobile Comput.*, vol. 16, no. 9, pp. 2392–2407, Sep. 2017.

[17] N. R. Devanur and J. D. Hartline, "Limited and online supply and the bayesian foundations of prior-free mechanism design," in *Proc. 10th ACM Conf. Electron. Commerce*, 2009, pp. 41–50.

[18] N. R. Devanur *et al.*, "Envy freedom and prior-free mechanism design," *J. Econ. Theory*, vol. 156, pp. 103–143, 2015.

[19] N. C. Luong, Z. Xiong, P. Wang, and D. Niyato, "Optimal auction for edge computing resource management in mobile blockchain networks: A deep learning approach," in *Proc. IEEE Int. Conf. Commun.*, 2018, pp. 1–6.

[20] J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5031–5044, May 2019.

[21] C. Liu, M. Bennis, M. Debbah, and H. V. Poor, "Dynamic task offloading and resource allocation for ultra-reliable low-latency edge computing," *IEEE Trans. Commun.*, vol. 67, no. 6, pp. 4132–4150, Jun. 2019.

[22] E. El Haber, T. M. Nguyen, and C. Assi, "Joint optimization of computational cost and devices energy for task offloading in multi-tier edge-clouds," *IEEE Trans. Commun.*, vol. 67, no. 5, pp. 3407–3421, May 2019.

[23] W. Sun, J. Liu, Y. Yue, and P. Wang, "Joint resource allocation and incentive design for blockchain-based mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 9, pp. 6050–6064, Sep. 2020.

[24] D. Zhang *et al.*, "Near-optimal and truthful online auction for computation offloading in green edge-computing systems," *IEEE Trans. Mobile Comput.*, vol. 19, no. 4, pp. 880–893, Apr. 2020.

[25] C. Wang, M. D. Renzo, S. Stanczak, S. Wang, and E. G. Larsson, "Artificial intelligence enabled wireless networking for 5G and beyond: Recent advances and future challenges," *IEEE Wireless Commun.*, vol. 27, no. 1, pp. 16–23, Feb. 2020.

[26] Z. M. Fadlullah *et al.*, "State-of-the-art deep learning: Evolving machine intelligence toward tomorrows intelligent network traffic control systems," *IEEE Commun. Surv. Tut.*, vol. 19, no. 4, pp. 243–255, Oct.–Dec. 2017.

[27] N. Kato *et al.*, "The deep learning vision for heterogeneous network traffic control: Proposal, challenges, and future perspective," *IEEE Wireless Commun.*, vol. 24, no. 3, pp. 146–53, Jun. 2017.

[28] T. Q. Dinh, Q. D. La, T. Q. S. Quek, and H. Shin, "Learning for computation offloading in mobile edge computing," *IEEE Trans. Commun.*, vol. 66, no. 12, pp. 6353–6367, Dec. 2018.

[29] Z. Xiong, Y. Zhang, D. Niyato, R. Deng, P. Wang, and L. Wang, "Deep reinforcement learning for mobile 5G and beyond: Fundamentals, applications, and challenges," *IEEE Veh. Technol. Mag.*, vol. 14, no. 2, pp. 44–52, Jun. 2019.

[30] C. Liu *et al.*, "A new deep learning-based food recognition system for dietary assessment on an edge computing service infrastructure," *IEEE Trans. Serv. Comput.*, vol. 11. no. 2, pp. 249–61, Mar. 2018.

[31] Y. Liu, H. Yu, S. Xie, and Y. Zhang, "Deep reinforcement learning for offloading and resource allocation in vehicle edge computing and networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 11158–11168, Nov. 2019.

[32] L. Huang, S. Bi, and Y. J. A. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Trans. Mobile Comput.*, vol. 19, no. 11, pp. 2581–2593, Nov. 2020.

[33] Y. He, N. Zhao, and H. Yin, "Integrated networking, caching, and computing for connected vehicles: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 44–55, Jan. 2018.

[34] M. Hu, L. Zhuang, D. Wu, Y. Zhou, X. Chen, and L. Xiao, "Learning driven computation offloading for asymmetrically informed edge computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 8, pp. 1802–1815, Aug. 2019.

[35] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic Game Theory*. Cambridge, U.K.: Cambridge Univ. Press, 2007.

[36] A. Gopinathan, N. Carlsson, Z. Li, and C. Wu, "Revenue-maximizing and truthful online auctions for dynamic spectrum access," in *Proc. Conf. Wireless On-demand Netw. Syst. Serv.*, 2016, pp. 1–8.

[37] B. Fan, H. Tian, L. Jiang, and A. V. Vasilakos, "A social-aware virtual MAC protocol for energy-efficient D2D communications underlying heterogeneous cellular networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8372–8385, Sep. 2018.

[38] Y. Wen, W. Zhang, and H. Luo, "Energy-optimal mobile application execution: Taming resource-poor mobile devices with cloud clones," in *Proc. IEEE INFOCOM*, 2012, pp. 2716–2720.

[39] R. B. Myerson, "Optimal auction design" *Math. Oper. Res.*, vol. 6, pp. 58–73, 1981.

[40] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2014.

[41] D. P. Bertsekas, *Constrained Optimization and Lagrange Multipliser Methods*. New York, NY, USA: Academic Press, 2014.

[42] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang, "Computation offloading and resource allocation in wireless cellular networks with mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 16, no. 8, pp. 4924–4938, Aug. 2017.

[43] C. You, K. Huang, H. Chae, and B. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.

[44] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.

[45] E. Li, L. Zeng, Z. Zhou, and X. Chen, "Edge AI: On-demand accelerating deep neural network inference via edge computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 447–457, Jan. 2020.

Gang Li (Student Member, IEEE) received the M.S. degree in information and communication systems from the Guilin University of Electronic Technology, Guilin, China, in 2016. He is currently working toward the Ph.D. degree with the Department of Electrical and Computer Engineering, Concordia University, Montreal, QC, Canada. His current research interests include mobile edge computing, cooperative communications, online algorithm, mechanism design, and machine learning. He was awarded the International Graduate Student Scholarship from the University of Manitoba in 2018 and was the recipient of the Concordia International Tuition Award of Excellence for 2019-2020.

Jun Cai (Senior Member, IEEE) received the Ph.D. degree from the University of Waterloo, Waterloo, ON, Canada, in 2004. From June 2004 to April 2006, he was with McMaster University, Hamilton, ON, Canada, as a Natural Sciences and Engineering Research Council of Canada (NSERC) Postdoctoral Fellow. From July 2006 to December 2018, he was with the Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, MB, Canada, where he was a Full Professor and the NSERC Industrial Research Chair. Since January 2019, he has been joined the Department of Electrical and Computer Engineering, Concordia University, Montreal, QC, Canada, as a Full Professor and the PERFORM Centre Research Chair. His current research interests include edge/fog computing, ehealth, radio resource management in wireless communication networks, and performance analysis. Dr. Cai was the Technical Program Committee (TPC) Co-Chair for IEEE GreenCom 2018, the Track/Symposium TPC Co-Chair for the IEEE VTC-Fall 2020, 2019, 2012, IEEE CCECE 2017, IEEE Globecom 2010, and IWCMC 2008, the Publicity Co-Chair for IWCMC 2010, 2011, 2013, 2014, 2015, 2017, 2020, and the Registration Chair for QShine 2005. He was also served on the Editorial Board of the IEEE INTERNET OF THINGS JOURNAL, *IET Communications*, and *Wireless Communications and Mobile Computing*. He was the recipient of the Best Paper Award from Chinacom in 2013, the Rh Award for outstanding contributions to research in applied sciences in 2012 from the University of Manitoba, and the Outstanding Service Award from IEEE Globecom 2010.

Shuang Ni received the B.E. degree in electronic information engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2017, and the M.S. degree in electrical and computer engineering from Concordia University, Montreal, QC, Canada, in 2021. She is currently working toward the Ph.D. degree in computer science with the University of Montreal, Montreal, QC, Canada. Her research interests include machine learning and topological data analysis, especially in biomedical field.