Reconfigurable Intelligent Surface Aided Mobile Edge Computing: From Optimization-Based to Location-Only Learning-Based Solutions

Xiaoyan Hu, *Member, IEEE,* Christos Masouros, *Senior Member, IEEE,* Kai-Kit Wong, *Fellow, IEEE*

Abstract—In this paper, we explore optimization-based and data-driven solutions in a reconfigurable intelligent surface (RIS)-aided multi-user mobile edge computing (MEC) system, where the user equipment (UEs) can partially offload their computation tasks to the access point (AP). We aim at maximizing the total completed task-input bits (TCTB) of all UEs with limited energy budgets during a given time slot, through jointly optimizing the RIS reflecting coefficients, the AP's receive beamforming vectors, and the UEs' energy partition strategies for local computing and offloading. A three-step block coordinate descending (BCD) algorithm is first proposed to effectively solve the non-convex TCTB maximization problem with guaranteed convergence. In order to reduce the computational complexity and facilitate lightweight online implementation of the optimization algorithm, we further construct two deep learning architectures. The first one takes channel state information (CSI) as input, while the second one exploits the UEs' locations only for online inference. The two data-driven approaches are trained using data samples generated by the BCD algorithm via supervised learning. Our simulation results reveal a close match between the performance of the optimization-based BCD algorithm and the low-complexity learning-based architectures, all with superior performance to existing schemes in both cases with perfect and imperfect input features. Importantly, the location-only deep learning method is shown to offer a particularly practical and robust solution alleviating the need for CSI estimation and feedback when line-of-sight (LoS) direct links exist between UEs and the AP.

Index Terms—Mobile edge computing, reconfigurable intelligent surface, receive beamforming, energy partition, deep learning.

I. INTRODUCTION

A. Motivations and Prior Works

The increasing data rates provided by 5G and beyond technologies, together with the proliferation of Internet-of-things (IoT) devices, have recently given rise to massive connectivity communications. Accompanied by a wide range of emerging computation-intensive applications, the computing and processing demands for user equipment (UEs), e.g., smart devices and IoT sensors, are growing unprecedentedly. In order to liberate the resource-limited UEs from heavy computation workloads and provide them with high-performance low-latency computing services, mobile edge computing (MEC) promotes to use cloud computing capabilities at the edge of mobile networks through integrating MEC servers at the

wireless access points (APs) [2]. Hence, UEs' computation-intensive tasks can be offloaded and completed at the adjacent APs with less cost, energy and time.

Extensive works have contributed to the performance enhancement of applying MEC in various wireless networks, either improving the energy efficiency or reducing the execution latency through jointly optimizing the radio and computational resources [3–8]. A multicell MEC system was considered in [3], where users' energy consumption was minimized through joint resource allocation. Later in [4], a game-theoretic algorithm was proposed to maximize the cell load as well as minimize the cost of time and energy. The offloading priority function was defined in [5] to show the relationship between the offloading strategy and resource allocation. A wireless powered MEC system was investigated in [6], where user cooperation was utilized to counteract the double-near-far effect. Work in [7] addressed the joint resource allocation of a multi-user multi-server MEC scenario to maximize the system utility. The complementary benefits of edge and central cloud computing were studied in a two-tier heterogeneous cloud computing network in [8].

In order to further enhance the uplink offloading performance of the resource-limited UEs, great attentions have been drawn to the technology of reconfigurable intelligent surface (RIS) recently, due to its advantages of low cost, easy deployment, fine-grained passive beamforming and directional signal enhancement or interference nulling [9–11]. Through controlling the reflecting elements on the surface, RISs can be reconfigured to provide a more favourable wireless propagation environment for communications. Clearly, leveraging RISs into MEC systems is a cost-effective and environment-friendly way to facilitate UEs' computation offloading.

Several pioneer RIS-aided MEC works have been done to explore the potential benefits of utilizing RISs in MEC systems [12–15]. A multi-user RIS-aided MEC system was considered in [12], where the execution latency was minimized with joint optimization on resource allocation and RIS coefficients design in an iterative way. It was verified that significant performance improvement can be attained compared to the case without RIS. The advantages of RIS in directional beamforming were exploited for both uplink task offloading and downlink results downloading in [13], where the power minimization problem was solved with an iterative block-structured algorithm. Similarly, both uplink and downlink transmissions were considered in the RIS-aided MEC work [14], while the system utility was maximized iteratively to reduce the cost of energy and

time. Later in [15], RIS was used in a wireless powered MEC system, and the energy consumption was minimized through a two-step iterative method.

For RIS-aided MEC systems, the formulated performance enhancement problems are typically non-convex with coupled optimization variables. Hence, iterative algorithms are usually necessary for jointly optimizing the radio and computational resource allocation as well as the RIS coefficients design. It is true that iterative algorithms may be capable of providing near-optimal solution even with guaranteed convergence, but they have very high computational complexity which require long execution time and thus may hinder their utilization in practical networks. To tackle this issue, deep learning architectures provide a promising way to achieve lightweight online implementation via offline training [16–18].

Note that deep learning methods have been investigated in some MEC systems to simplify the optimization algorithm or fulfill online implementations [19–22]. A deep learning-based offloading strategy was designed in [20] to minimize the weighted energy consumption and latency. The deep reinforcement learning (DRL) was leveraged in [21] for smart resource allocation of a software defined network (SDN)-enabled MEC architecture, also in [22] for online offloading decisions and resource allocation of a wireless powered MEC system. Recently, the DRL was also used in the RIS-aided architectures to enhance the security [23] and maximize the sum rate of the downlink communications [24]. In [25], a convolutional neural network (CNN) was constructed for channel estimation of a large RIS-aided millimeter-wave (mm-wave) communication system.

B. Our Contributions

As per the above literature, deep learning approaches are promising to offer low-complexity solutions for the traditional MEC-related systems or RIS-aided downlink communication architectures. However, the potentials of deep learning methods in simplifying the optimization algorithms of complex RIS-aided MEC systems have not been explored in the existing literature. In this paper, a multi-user RIS-aided MEC architecture with multiplexing computation offloading is considered, where the RIS is installed to constructively control the interference and enhance the overall performance of UEs. We not only propose an iterative optimization algorithm to efficiently solve the formulated problem with guaranteed convergence, but also construct two deep learning architectures to facilitate online implementations of the proposed algorithm with significantly reduced complexity. To the best of our knowledge, this is the first work that leverages the data-driven approach in the RIS-aided MEC systems. Also, the proposed optimization-based algorithm is used to train the deep neural networks (DNNs) for efficient optimization and lightweight online implementation.

Our main contributions are summarized as follows:

- *A RIS-aided MEC architecture with uplink multiplexing offloading is leveraged to enhance the performance for maximizing the total completed task-input bits (TCTB) of all the resource-limited UEs.* Partial computation offloading is adopted for the RIS-aided MEC system in a multiplexing way, where the UEs can partially offload

their computation task-input bits simultaneously. We aim at maximizing the TCTB of all the UEs with limited energy supply budgets during a given time slot, which maximizes the computation efficiency in both time and energy, through jointly optimizing the RIS reflecting coefficients, the AP's receive beamforming vectors, and the UEs' energy partition strategies for local computing and computation offloading. The utilization of RIS is capable to enhance the performance of maximizing TCTB by constructively reconfiguring favourable propagations for all the UEs.

- *The RIS reflecting coefficients design, receive beamforming design, and energy partition optimization for maximizing the TCTB are effectively addressed through a three-step block coordinate descending (BCD) optimization algorithm with guaranteed convergence.* The non-convex property and strongly coupled optimization variables of the formulated TCTB maximization problem make it difficult to obtain the global optimal solution. To address this issue, we propose a three-step BCD optimization algorithm to effectively separate the coupling and solve the problem by addressing three sub-problems iteratively. The DC (difference of convex functions) programming method is leveraged to solve the first and third sub-problems respectively for RIS reflecting coefficients design and receive beamforming design with guaranteed convergence, while the optimal solution of the second sub-problem for receive beamforming design is obtained via eigenvalue decompositions.

- *A CSI-based deep learning architecture with DNN-CSI is constructed to facilitate the online implementation of the proposed optimization-based BCD algorithm with significantly reduced complexity.* Supervised learning is adopted to train the DNN-CSI using the data samples generated by the proposed BCD algorithm. It is shown that this CSI-based data-driven method can sufficiently capture the mapping of the BCD algorithm to the optimization solution, with lightweight inference complexity. Satisfactory and stable performance can be achieved in both scenarios without and with strong line-of-sight (LoS) direct links between the UEs and the AP.

- *A location-only deep learning architecture is further constructed that can effectively predict the solution of the proposed BCD algorithm without the need for pilot channel estimation and feedback during online inference.* This data-driven method is also based on supervised learning, and it performs well when LoS direct links are available for UEs. The complexity of both training and testing can be greatly reduced compared with the CSI-based learning architecture since only UEs' locations are needed as the input feature. Thus, more lightweight online implementation can be achieved.

- *High effectiveness and robustness of the CSI-based and the location-only deep learning architectures to the uncertainty of the input CSI and UEs' locations are validated.* The scenarios with corrupted input features to the two proposed deep learning architectures are considered to validate their effectiveness and robustness.
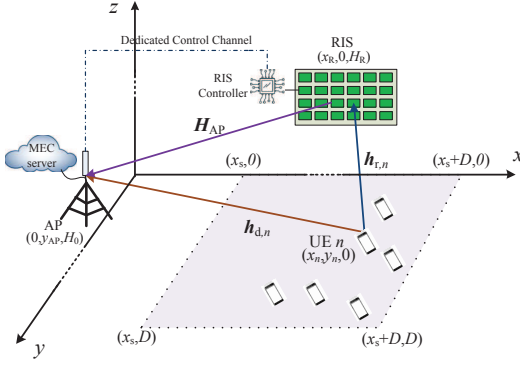
Fig. 1. An illustration of the RIS-aided MEC architecture, where a K-element RIS assists the multiplexing computation offloading of N UEs. The phase shifts of the RIS elements can be adaptively adjusted by the AP through the dedicated control channel, so as to refine the signal propagations.

Simulation results are presented to evaluate the performance of the proposed BCD optimization algorithm and the two deep learning architectures. It is confirmed that the proposed BCD algorithm highly outperforms the other three traditional benchmarks. In addition, the CSI-based deep learning architecture can always approach the performance of the BCD algorithm in both scenarios without and with LoS direct links between the UEs and the AP. It is noticeable that the location-only deep learning architecture can replace the CSI-based architecture to provide a satisfactory data-driven solution in the scenario with LoS direct links, with much less required overheads. Besides, it is shown that the constructed two deep learning architectures can effectively emulating the proposed BCD algorithm even in the more practical scenarios with uncertainty in the input information of CSI and UEs' locations.

The rest of this paper is organized as follows. Section II introduces the considered system model and presents the corresponding problem formulation. The BCD algorithm is proposed in Section III to provide an optimization solution to the formulated problem, while two deep learning architectures are shown in Section IV to offer the learning-based solutions. The implementation setting and complexity reduction of the deep learning approaches are discussed in Section V. Section VI provides the simulation results, and we conclude our paper in Section VII.

Notations—In this paper, the upper and lower case bold symbols denote matrices and vectors, respectively. The notations $(\cdot)^{\mathrm{T}}$ and $(\cdot)^{\mathrm{H}}$ represent transpose and conjugate transpose for vectors or matrices. In addition, $\otimes$ denotes the Kronecker product. $\mathrm{Tr}\{\mathbf{A}\}$ is the trace of square matrix $\mathbf{A}$. Also, $\mathrm{eig}\{\mathbf{A}\}$ denotes the set of all the eigenvalues of $\mathbf{A}$, and $\mathrm{eigvec}\{\cdot\}$ gives the eigenvector for a given eigenvalue of $\mathbf{A}$. $\nabla_{\mathbf{X}} f(\mathbf{X})$ denotes the Jacobian matrix of function $f(\mathbf{X})$ with respect to (w.r.t.) the matrix $\mathbf{X}$, and $\partial_{\mathbf{X}} g(\mathbf{X})$ denotes a subgradient of function $g(\mathbf{X})$ w.r.t. $\mathbf{X}$. $\langle \mathbf{X}_1, \mathbf{X}_2 \rangle \triangleq \Re\{\mathrm{tr}(\mathbf{X}_1^{\mathrm{H}} \mathbf{X}_2)\}$, where $\Re\{\cdot\}$ is the real-value operator. Finally, $\mathrm{diag}\{\mathbf{x}\}$ is the diagonal matrix formed by the elements of vector $\mathbf{x}$.

II. SYSTEM MODEL AND PROBLEM FORMULATION

We consider a RIS-aided MEC system as shown in Fig. 1, which consists of N single-antenna ground UEs, one RIS with

K reflecting elements, and one M-antenna AP. The RIS can be flexibly installed on the surrounding building walls, and it is under the control of the AP through a wireless controller to dynamically adjust the phase shift of each reflecting element. By choosing a desirable location of the RIS, it is possible to achieve LoS connections between the RIS and the AP as well as the UEs within a certain area.

A. Completed Task-Input Bits (CTB) with Partial Offloading

Each UE $n \in \mathcal{N} = \{1, 2, \ldots, N\}$ has intensive computation task-input bits (e.g., program codes and input parameters) to be dealt with, but with a limited energy budget dedicated for completing these task bits, denoted as E_n in Joules (J). A partial offloading mode is adopted to handle UEs' computation tasks, which are suitable to the augmented reality (AR) applications mentioned in [2]. For these kinds of computation tasks, UEs' task-input bits can be arbitrarily divided to facilitate parallel operations at UEs for local computing and offloading to the AP for remote computing. Hence, the accounted computation energy consumption of each UE includes that both used for local computing and computation offloading. The grid-powered AP is co-located with a powerful MEC server for helping UEs compute their offloaded tasks and it is also capable of downloading UEs' computation results, both in negligible time.[1] We use C_n to represent the amount of required computing resource, i.e., the number of CPU cycles, for completing 1-bit of UE n's input data. Our aim is to maximize the TCTB of all UEs each with a limited energy budget during a given time slot T, which is equivalent to maximizing the computation efficiency of the RIS-assisted MEC system, including both energy efficiency and time efficiency referring to computation rate [26–29].

We first introduce a partition parameter $a_n \in [0, 1]$ for UE $n \in \mathcal{N}$, and $a_n E_n$ J of energy will be used for computation offloading while $(1 - a_n) E_n$ J of energy will be used for local computing. In this case, the transmit power of UE n for computation offloading is given as

$$p_n = \frac{a_n E_n}{T} \triangleq a_n \widetilde{E}_n, \ \forall n \in \mathcal{N}, \tag{1}$$

with $\widetilde{E}_n = E_n/T$ for $n \in \mathcal{N}$.

Let s_n denote the task-input data-bearing signal transmitted by UE $n \in \mathcal{N}$ for computation offloading with $|s_n| = 1$. Note that all the UEs with offloading requirements transmit their signals simultaneously in a multiplexing way within the given time slot, and thus we can express the corresponding received signal $\mathbf{y} \in \mathbb{C}^{M \times 1}$ at the AP as [31]

$$\mathbf{y} = \sum_{n=1}^{N} (\mathbf{H}_{\mathrm{AP}} \boldsymbol{\Phi} \mathbf{h}_{\mathrm{r},n} + \mathbf{h}_{\mathrm{d},n}) \sqrt{p_n} s_n + \mathbf{n}, \tag{2}$$

---

[1]In this paper, we ignore the execution latency at the MEC server due to the fact that the MEC server is grid-power supplied and has a super-high computing capability, and thus the corresponding execution latency is negligible compared with that consumed at the UEs [5, 6, 8, 15, 26–30]. In addition, we assume that the UEs' computation results are with very small sizes, e.g., a few command bits, which can be ignored especially compared with their task-input bits, and thus the AP with a sufficient power supply can transmit the results back to UEs with negligible time [5–8, 12, 14, 15, 26–28].

where $\mathbf{h}_{\mathrm{d},n} \in \mathbb{C}^{M \times 1}$ is the direct link between UE n and the AP, $\mathbf{h}_{\mathrm{r},n} \in \mathbb{C}^{K \times 1}$ indicates the relay channel between UE n and the RIS, and $\mathbf{H}_{\mathrm{AP}} \in \mathbb{C}^{M \times K}$ represents the channel between RIS and the AP. We assume that the channels $\{\mathbf{h}_{\mathrm{d},n}\}$, $\{\mathbf{h}_{\mathrm{r},n}\}$ and $\mathbf{H}_{\mathrm{AP}}$ are quasi-static within the given time slot. Additionally, $\mathbf{\Phi} = \mathrm{diag}\{\boldsymbol{\phi}\}$ indicates the reflection-coefficient matrix of the RIS, where $\boldsymbol{\phi} = [\phi_1, \ldots, \phi_K]^{\mathrm{T}}$ and $\phi_k = e^{j\theta_k}$ being the phase shift of the k-th reflecting element of the RIS with $\theta_k \in [0, 2\pi]$ for $k \in \mathcal{K} = \{1, 2, \ldots, K\}$. Also, $\mathbf{n} \sim \mathcal{CN}(0, \sigma^2 \mathbf{I}_M)$ is the the additive white Gaussian noise (AWGN) at the AP with σ^2 being the noise power. The linear beamforming strategy is then adopted at the AP to decode the UEs' transmit signals, and $\mathbf{w}_n \in \mathbb{C}^{M \times 1}$ is the specific receive beamforming vector for UE n. Thus, the estimated signal for UE n can be given as

$$\widehat{s}_n = \mathbf{w}_n^{\mathrm{H}} \mathbf{y} \tag{3}$$
$$= \mathbf{w}_n^{\mathrm{H}} \sum_{n=1}^{N} (\mathbf{H}_{\mathrm{AP}} \mathbf{\Phi} \mathbf{h}_{\mathrm{r},n} + \mathbf{h}_{\mathrm{d},n}) \sqrt{p_n} s_n + \mathbf{w}_n^{\mathrm{H}} \mathbf{n}, \ \forall n \in \mathcal{N}.$$

Based on the analysis above, we can obtain the uplink signal-to-interference-plus-noise ratio (SINR) for offloading UE n's tasks as

$$\gamma_n(\mathbf{a}, \mathbf{w}_n, \boldsymbol{\phi}) = \tag{4}$$
$$\frac{a_n \widetilde{E}_n |\mathbf{w}_n^{\mathrm{H}} (\mathbf{H}_{\mathrm{AP}} \mathbf{\Phi} \mathbf{h}_{\mathrm{r},n} + \mathbf{h}_{\mathrm{d},n})|^2}{\sum_{i=1, i \neq n}^{N} a_i \widetilde{E}_i |\mathbf{w}_n^{\mathrm{H}} (\mathbf{H}_{\mathrm{AP}} \mathbf{\Phi} \mathbf{h}_{\mathrm{r},i} + \mathbf{h}_{\mathrm{d},i})|^2 + \sigma^2 \|\mathbf{w}_n^{\mathrm{H}}\|^2}, \ \forall n \in \mathcal{N},$$

where we denote an energy partition vector $\mathbf{a} = [a_1, \ldots, a_N]$. Then, the CTB of UE n through computation offloading can be expressed as [32]

$$R_n^{\mathrm{off}}(\mathbf{a}, \mathbf{w}_n, \boldsymbol{\phi}) = BT \log_2(1 + \gamma_n(\mathbf{a}, \mathbf{w}_n, \boldsymbol{\phi})), \ \forall n \in \mathcal{N}, \tag{5}$$

where B is the given bandwidth shared with all the UEs.

As for the case of local computing, the dynamic voltage and frequency scaling (DVFS) technique is adopted at all the UEs for increasing the computation energy efficiency through adaptively controlling the CPU frequency used for computing [33]. Thus, the computation energy consumption of UE $n \in \mathcal{N}$ can be expressed as $T\kappa_n f_n^3$, where κ_n is the effective capacitance coefficient of UE n, and f_n is the CPU frequency of its processing server. Also, we have $(1 - a_n)E_n = T\kappa_n f_n^3$, and thus we can calculate f_n as

$$f_n = \sqrt[3]{\frac{(1 - a_n)E_n}{T\kappa_n}}, \ \forall n \in \mathcal{N}. \tag{6}$$

Hence, the CTB of UE n for local computing can be given as

$$R_n^{\mathrm{loc}}(a_n) = \frac{f_n T}{C_n} = \frac{T}{C_n} \sqrt[3]{\frac{(1 - a_n)\widetilde{E}_n}{\kappa_n}}, \ \forall n \in \mathcal{N}. \tag{7}$$

B. Problem Formulation

We aim at maximizing the TCTB of all the UEs with limited energy supply $\{E_n\}_{n \in \mathcal{N}}$ in the given time slot T, including their CTB through both computation offloading and local computing, where the objective TCTB is maximized through jointly optimizing the reflection coefficients in $\boldsymbol{\phi}$, the

receive beamforming vectors in $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_N]$, and the energy partition parameters in $\mathbf{a}$. As we mentioned before, maximizing the TCTB in this scenario can maximize the computation efficiency in both time and energy [26–29]. The corresponding TCTB maximization problem is formulated as problem (P0) given below

$$(\text{P0}): \max_{\mathbf{a}, \mathbf{W}, \boldsymbol{\phi}} \ \sum_{n=1}^{N} \left(R_n^{\mathrm{off}}(\mathbf{a}, \mathbf{w}_n, \boldsymbol{\phi}) + R_n^{\mathrm{loc}}(a_n) \right) \tag{8a}$$
$$\text{s.t.} \quad a_n \in [0, 1], \ \forall n \in \mathcal{N}, \tag{8b}$$
$$|\phi_n| = 1, \ \forall n \in \mathcal{N}, \tag{8c}$$

which is a non-convex optimization problem since the optimization variables $\boldsymbol{\phi}$, $\mathbf{W}$, and $\mathbf{a}$ are strongly coupled. Hence, it is computationally difficult to find the global optimal solution of problem (P0), similar in [12–15]. To address this issue, we propose a three-step BCD optimization algorithm to effectively separate the coupling among the optimization variables and solve this problem iteratively with guaranteed convergence.

III. BCD OPTIMIZATION ALGORITHM DESIGN

The proposed BCD algorithm is operated in three major steps by solving three sub-problems iteratively. In the χ-th ($\chi = 1, 2, \ldots$) iteration, we first design the RIS reflecting coefficients in $\boldsymbol{\phi}$, with given $\mathbf{W}_{\chi-1}$ and $\mathbf{a}_{\chi-1}$ obtained in the previous iteration, and denote the solution as $\boldsymbol{\phi}_{\chi}$. Then, with given $\mathbf{a}_{\chi-1}$ and the obtained $\boldsymbol{\phi}_{\chi}$, we show that the receive beamforming vectors in $\mathbf{W}$ have closed-form optimal solutions, presented as $\mathbf{W}_{\chi}$. We finally optimize the energy partition parameters in $\mathbf{a}$ with the obtained $\boldsymbol{\phi}_{\chi}$ and $\mathbf{W}_{\chi}$, indicating the solution as $\mathbf{a}_{\chi}$. With the given initial $\mathbf{W}_0$ and $\mathbf{a}_0$, we can prove that the proposed three-step BCD algorithm can be performed with guaranteed convergence. Next, we will demonstrate the details of the BCD optimization algorithm for solving the three sub-problems in the χ-th iteration.

A. RIS Reflecting Coefficients Design

In the χ-th iteration of the BCD algorithm, we first consider designing the RIS reflecting coefficients, i,e, $\boldsymbol{\phi}$, with given $\mathbf{W} = \mathbf{W}_{\chi-1}$ and $\mathbf{a} = \mathbf{a}_{\chi-1}$. With the given energy partition parameters in $\mathbf{a}$, we can equivalently obtain the UEs' transmit power as $p_n = a_n \widetilde{E}_n$. Then, the RIS's reflecting coefficients design problem (P1) can be described as

$$(\text{P1}): \max_{\boldsymbol{\phi}} \ \sum_{n=1}^{N} \log_2(1 + \gamma_n(\boldsymbol{\phi})) \tag{9a}$$
$$\text{s.t.} \quad |\phi_k| = 1, \ \forall k \in \mathcal{K}, \tag{9b}$$

which is still non-convex and difficult to deal with directly. According to the expression of $\gamma_n(\boldsymbol{\phi})$ in (4) for $n \in \mathcal{N}$, we can re-express $|\mathbf{w}_n^{\mathrm{H}}(\mathbf{H}_{\mathrm{AP}} \mathbf{\Phi} \mathbf{h}_{\mathrm{r},i} + \mathbf{h}_{\mathrm{d},i})|^2$ as

$$|\mathbf{w}_n^{\mathrm{H}}(\mathbf{H}_{\mathrm{AP}} \mathbf{\Phi} \mathbf{h}_{\mathrm{r},i} + \mathbf{h}_{\mathrm{d},i})|^2 \tag{10}$$
$$= |\mathbf{w}_n^{\mathrm{H}} \mathbf{H}_{\mathrm{AP}} \mathrm{diag}(\mathbf{h}_{\mathrm{r},i}) \boldsymbol{\phi} + \mathbf{w}_n^{\mathrm{H}} \mathbf{h}_{\mathrm{d},i}|^2$$
$$= |\mathbf{h}_{\mathrm{r},n,i}^{\mathrm{RIS}} \boldsymbol{\phi} + h_{\mathrm{d},n,i}|^2, \ \forall n, i \in \mathcal{N},$$

where $\mathbf{h}_{\mathrm{r},n,i}^{\mathrm{RIS}} = \mathbf{w}_n^{\mathrm{H}}\mathbf{H}_{\mathrm{AP}}\mathrm{diag}(\mathbf{h}_{\mathrm{r},i}) \in \mathbb{C}^{1\times K}$ and $h_{\mathrm{d},n,i} = \mathbf{w}_n^{\mathrm{H}}\mathbf{h}_{\mathrm{d},i}$. By defining a matrix $\mathbf{Q}_{n,i} \in \mathbb{C}^{(K+1)\times(K+1)}$ as

$$\mathbf{Q}_{n,i} = \begin{bmatrix} (\mathbf{h}_{\mathrm{r},n,i}^{\mathrm{RIS}})^{\mathrm{H}}\mathbf{h}_{\mathrm{r},n,i}^{\mathrm{RIS}}, & (\mathbf{h}_{\mathrm{r},n,i}^{\mathrm{RIS}})^{\mathrm{H}}h_{\mathrm{d},n,i} \\ h_{\mathrm{d},n,i}^{\mathrm{H}}\mathbf{h}_{\mathrm{r},n,i}^{\mathrm{RIS}}, & 0 \end{bmatrix}, \ \forall n \in \mathcal{N}, \quad (11)$$

and a vector $\widetilde{\boldsymbol{\phi}} = [\phi_1,\ldots,\phi_K,\xi]^{\mathrm{T}} \in \mathbb{C}^{(K+1)\times 1}$ with an auxiliary scalar ξ, we can then re-express $|\mathbf{h}_{\mathrm{r},n,i}^{\mathrm{RIS}}\boldsymbol{\phi}+h_{\mathrm{d},n,i}|^2 = \widetilde{\boldsymbol{\phi}}^{\mathrm{H}}\mathbf{Q}_{n,i}\widetilde{\boldsymbol{\phi}} + |h_{\mathrm{d},n,i}|^2 = \mathrm{Tr}(\mathbf{Q}_{n,i}\boldsymbol{\Psi}) + |h_{\mathrm{d},n,i}|^2$, where $\boldsymbol{\Psi} = \widetilde{\boldsymbol{\phi}}\widetilde{\boldsymbol{\phi}}^{\mathrm{H}} \in \mathbb{C}^{(K+1)\times(K+1)}$ is a positive semidefinite matrix (PSD) related to the RIS reflecting coefficients.

Note that each added item in the objective function, i.e., $\log_2(1+\gamma_n(\boldsymbol{\phi}))$, can be re-written as

$$\log_2(1+\gamma_n(\boldsymbol{\phi})) = \log_2(1+\gamma_n(\widetilde{\boldsymbol{\phi}})) \quad (12)$$

$$= \log_2\left(\sum_{j=1}^{N} p_j|\mathbf{w}_n^{\mathrm{H}}(\mathbf{H}_{\mathrm{AP}}\boldsymbol{\Phi}\mathbf{h}_{\mathrm{r},j}+\mathbf{h}_{\mathrm{d},j})|^2 + \sigma^2\|\mathbf{w}_n^{\mathrm{H}}\|^2\right)$$

$$- \log_2\left(\sum_{i=1,i\neq n}^{N} p_i|\mathbf{w}_n^{\mathrm{H}}(\mathbf{H}_{\mathrm{AP}}\boldsymbol{\Phi}\mathbf{h}_{\mathrm{r},i}+\mathbf{h}_{\mathrm{d},i})|^2 + \sigma^2\|\mathbf{w}_n^{\mathrm{H}}\|^2\right)$$

$$= \log_2\left(\sum_{j=1}^{N} p_j(\mathrm{Tr}(\mathbf{Q}_{n,j}\boldsymbol{\Psi}) + |h_{\mathrm{d},n,j}|^2) + \sigma^2\|\mathbf{w}_n^{\mathrm{H}}\|^2\right)$$

$$- \log_2\left(\sum_{i=1,i\neq n}^{N} p_i(\mathrm{Tr}(\mathbf{Q}_{n,i}\boldsymbol{\Psi}) + |h_{\mathrm{d},n,i}|^2) + \sigma^2\|\mathbf{w}_n^{\mathrm{H}}\|^2\right)$$

$$\triangleq F_{1,n}(\boldsymbol{\Psi}) - F_{2,n}(\boldsymbol{\Psi}), \ \forall n \in \mathcal{N},$$

where $F_{1,n}(\boldsymbol{\Psi})$ and $F_{2,n}(\boldsymbol{\Psi})$ are two concave functions w.r.t. $\boldsymbol{\Psi}$. Hence, the problem (P1) can be equivalently transformed into the following problem $(\widetilde{\mathrm{P}}1)$

$$(\widetilde{\mathrm{P}}1) : \max_{\boldsymbol{\Psi}\succeq\mathbf{0}} \ \sum_{n=1}^{N} F_{1,n}(\boldsymbol{\Psi}) - F_{2,n}(\boldsymbol{\Psi}) \quad (13a)$$

$$\text{s.t.} \quad \boldsymbol{\Psi}_{k,k} = 1, \ \forall k = 1,2,\ldots,K+1, \quad (13b)$$

$$\mathrm{rank}(\boldsymbol{\Psi}) = 1. \quad (13c)$$

Even though the objective function in (13a) and the rank-one constraint (13b) make problem $(\widetilde{\mathrm{P}}1)$ non-convex, it is easy to note that the objective function is a sum of differences of concave functions. Next, we will show that the DC programming [34] can be leveraged to effectively address the issues of the objective function and the rank-one constraint.

As for the objective function, in the $(l+1)$-th $(l = 0,1,\ldots)$ iteration of the DC programming, the second concave item, i.e., $F_{2,n}(\boldsymbol{\Psi})$ for $n \in \mathcal{N}$, can be approximated by its linear upper bound at the point $\boldsymbol{\Psi}^{(l)}$ (the solution obtained from the previous l-th iteration), which is given as

$$F_{2,n}(\boldsymbol{\Psi}) \leq \widehat{F}_{2,n}(\boldsymbol{\Psi};\boldsymbol{\Psi}^{(l)}) = F_{2,n}(\boldsymbol{\Psi}^{(l)}) + \quad (14)$$

$$\frac{\sum_{i=1,i\neq n}^{N} p_i\left\langle(\boldsymbol{\Psi}-\boldsymbol{\Psi}^{(l)}), \nabla_{\boldsymbol{\Psi}}\mathrm{Tr}(\mathbf{Q}_{n,i}\boldsymbol{\Psi})|_{\boldsymbol{\Psi}=\boldsymbol{\Psi}^{(l)}}\right\rangle}{\ln 2\left(\sum_{i=1,i\neq n}^{N} p_i(\mathrm{Tr}(\mathbf{Q}_{n,i}\boldsymbol{\Psi}^{(l)}) + |h_{\mathrm{d},n,i}|^2) + \sigma^2\|\mathbf{w}_n^{\mathrm{H}}\|^2\right)},$$

where $\nabla_{\boldsymbol{\Psi}}\mathrm{Tr}(\mathbf{Q}_{n,i}\boldsymbol{\Psi})|_{\boldsymbol{\Psi}=\boldsymbol{\Psi}^{(l)}}$ denotes the Jacobian matrix of $\mathrm{Tr}(\mathbf{Q}_{n,i}\boldsymbol{\Psi})$ w.r.t. $\boldsymbol{\Psi}$ at the point $\boldsymbol{\Psi}^{(l)}$, and it is easy to note that the equality holds when $\boldsymbol{\Psi} = \boldsymbol{\Psi}^{(l)}$.

As for the rank-one constraint, it can be equivalentlly transformed into the following form

$$\mathrm{Tr}(\boldsymbol{\Psi}) - \|\boldsymbol{\Psi}\|_{\mathrm{s}} = 0, \quad (15)$$

where $\|\boldsymbol{\Psi}\|_{\mathrm{s}}$ denotes the spectral norm of the PSD matrix $\boldsymbol{\Psi}$. It is noticeable that $\mathrm{Tr}(\boldsymbol{\Psi}) = \sum_{k=1}^{K+1}\rho_k(\boldsymbol{\Psi})$ and $\|\boldsymbol{\Psi}\|_{\mathrm{s}} = \rho_1(\boldsymbol{\Psi})$, where $\rho_k(\boldsymbol{\Psi})$ indicates the k-th largest singular value of $\boldsymbol{\Psi}$. Hence, the equality of $\mathrm{Tr}(\boldsymbol{\Psi}) = \|\boldsymbol{\Psi}\|_{\mathrm{s}}$ holds when the rank-one constraint is satisfied with $\rho_1(\boldsymbol{\Psi}) > 0$ and $\rho_k(\boldsymbol{\Psi}) = 0$ for $k = 2,\ldots,K+1$, and vice versa. Similarly, in the $(l+1)$-th iteration of the DC programming, a linear lower-bound of the convex item $\|\boldsymbol{\Psi}\|_{\mathrm{s}}$ at the point $\boldsymbol{\Psi}^{(l)}$ can be expressed as

$$\|\boldsymbol{\Psi}\|_{\mathrm{s}} \geq \|\boldsymbol{\Psi}^{(l)}\|_{\mathrm{s}} + \left\langle(\boldsymbol{\Psi}-\boldsymbol{\Psi}^{(l)}), \partial_{\boldsymbol{\Psi}}\|\boldsymbol{\Psi}\|_{\mathrm{s}}|_{\boldsymbol{\Psi}=\boldsymbol{\Psi}^{(l)}}\right\rangle \quad (16)$$

$$\triangleq \Upsilon(\boldsymbol{\Psi};\boldsymbol{\Psi}^{(l)}),$$

where $\partial_{\boldsymbol{\Psi}}\|\boldsymbol{\Psi}\|_{\mathrm{s}}|_{\boldsymbol{\Psi}=\boldsymbol{\Psi}^{(l)}}$ is a subgradient of the spectral norm $\|\boldsymbol{\Psi}\|_{\mathrm{s}}$ w.r.t. $\boldsymbol{\Psi}$ at the point $\boldsymbol{\Psi}^{(l)}$, and the equality holds when $\boldsymbol{\Psi} = \boldsymbol{\Psi}^{(l)}$. Note that one subgradient of $\|\boldsymbol{\Psi}\|_{\mathrm{s}}$ at point $\boldsymbol{\Psi}^{(l)}$ can be efficiently computed as $\mathbf{z}_1\mathbf{z}_1^{\mathrm{H}}$, where $\mathbf{z}_1$ is the vector corresponding to the largest singular value of $\boldsymbol{\Psi}^{(l)}$ [35].

With the obtained linear lower bound of $\|\boldsymbol{\Psi}\|_{\mathrm{s}}$ in (16), we can generate an approximate rank-one constraint of (15), which is shown as

$$\mathrm{Tr}(\boldsymbol{\Psi}) - \Upsilon(\boldsymbol{\Psi};\boldsymbol{\Psi}^{(l)}) \leq \varepsilon_{\boldsymbol{\Psi}}, \quad (17)$$

where $\varepsilon_{\boldsymbol{\Psi}}$ is a positive threshold with very small value close to zero. The approximated rank-one constraint can guarantee that $0 \leq \mathrm{Tr}(\boldsymbol{\Psi}) - \|\boldsymbol{\Psi}\|_{\mathrm{s}} \leq \mathrm{Tr}(\boldsymbol{\Psi}) - \Upsilon(\boldsymbol{\Psi};\boldsymbol{\Psi}^{(l)}) \leq \varepsilon_{\boldsymbol{\Psi}}$, and the rank-one constraint can be approached with an arbitrary accuracy by setting $\varepsilon_{\boldsymbol{\Psi}}$ infinitely close to zero.

Hence, we can obtain an approximation problem of $(\widetilde{\mathrm{P}}1)$ at the $(l+1)$-th iteration as

$$(\mathrm{P}1.1) : \max_{\boldsymbol{\Psi}\succeq\mathbf{0}} \ \sum_{n=1}^{N} F_{1,n}(\boldsymbol{\Psi}) - \widehat{F}_{2,n}(\boldsymbol{\Psi};\boldsymbol{\Psi}^{(l)}) \quad (18a)$$

$$\text{s.t.} \quad \boldsymbol{\Psi}_{k,k} = 1, \ \forall k = 1,2,\ldots,K+1, \quad (18b)$$

$$\mathrm{Tr}(\boldsymbol{\Psi}) - \Upsilon(\boldsymbol{\Psi};\boldsymbol{\Psi}^{(l)}) \leq \varepsilon_{\boldsymbol{\Psi}}, \quad (18c)$$

which is a convex optimization problem and can be readily solved by the existing convex solvers such as CVX [36], and the optimal solution can be obtained as $\boldsymbol{\Psi}^{(l+1)}$. Through choosing $\boldsymbol{\Psi}^{(0)} = \boldsymbol{\Psi}_{\chi-1} = \widetilde{\boldsymbol{\phi}}_{\chi-1}\widetilde{\boldsymbol{\phi}}_{\chi-1}^{\mathrm{H}}$, it is easy to prove that the feasibility of problem (P1.1) in each iteration l can always be guaranteed since $\boldsymbol{\Psi}^{(l-1)}$ is always a feasible solution.

Lemma 1. *The objective function of problem (P1) in* (13a) *monotonically increases with the iteration index l as*

$$F_{1,n}(\boldsymbol{\Psi}^{(l+1)}) - F_{2,n}(\boldsymbol{\Psi}^{(l+1)}) \quad (19)$$

$$\overset{(a)}{\geq} F_{1,n}(\boldsymbol{\Psi}^{(l+1)}) - \widehat{F}_{2,n}(\boldsymbol{\Psi}^{(l+1)};\boldsymbol{\Psi}^{(l)})$$

$$\overset{(b)}{\geq} F_{1,n}(\boldsymbol{\Psi}^{(l)}) - \widehat{F}_{2,n}(\boldsymbol{\Psi}^{(l)};\boldsymbol{\Psi}^{(l)})$$

$$= F_{1,n}(\boldsymbol{\Psi}^{(l)}) - F_{2,n}(\boldsymbol{\Psi}^{(l)}), \ \forall n \in \mathcal{N},$$

where (a) comes from the inequality (14) *and (b) holds since $\boldsymbol{\Psi}^{(l)}$ is a feasible solution while $\boldsymbol{\Psi}^{(l+1)}$ is the optimal solution of problem (P1.1) in* (18). *Also, the objective function of*

problem $(\widetilde{P}1)$ is upper-bounded by the UEs' limited energy budgets. Hence, Problem $(\widetilde{P}1)$ in (13) as well as its equivalent form (P1) in (9) can be solved through the DC programming method with guaranteed convergence [34]. The final solution of $\boldsymbol{\Psi}$ at the convergence of the $(l+1)$-th iteration of the DC programming is the solution of the BCD algorithm at the χ-th iteration, i.e., $\boldsymbol{\Psi}_\chi = \boldsymbol{\Psi}^{(l+1)}$.

With the obtained $\boldsymbol{\Psi}_\chi$, we can retrieve $\widetilde{\phi}_\chi$ by decomposing $\boldsymbol{\Psi}_\chi = \widetilde{\phi}_\chi \widetilde{\phi}_\chi^{\mathrm{H}}$ with denoting $\widetilde{\phi}_\chi = [\phi_{\chi,0}, \xi_{\chi,0}]^{\mathrm{T}}$, and then it is easy to obtain the RIS reflecting coefficient vector at the χ-th iteration of the BCD algorithm as $\phi_\chi = \phi_{\chi,0}/\xi_{\chi,0}$ and accordingly $\boldsymbol{\Phi}_\chi = \mathrm{diag}\{\phi_\chi\}$. In order to facilitate the following analysis of designing the algorithm, we define the effective UE-AP channels with given $\boldsymbol{\Phi}$ (or ϕ) as

$$\mathbf{h}_n(\boldsymbol{\Phi}) = \mathbf{H}_{\mathrm{AP}} \boldsymbol{\Phi} \mathbf{h}_{\mathrm{r},n} + \mathbf{h}_{\mathrm{d},n}, \ \forall n \in \mathcal{N}. \qquad (20)$$

B. Receive Beamforming Design

With given $\mathbf{a} = \mathbf{a}_{\chi-1}$ ($p_n = a_n \widetilde{E}_n$, $n \in \mathcal{N}$) and $\boldsymbol{\Phi} = \boldsymbol{\Phi}_\chi$, the sub-problem for optimizing the AP's receive beamforming vextors for each UE, i,e, $\mathbf{w}_n$ for $n \in \mathcal{N}$, can be expressed as the following problem (P2)

$$(\text{P2}) : \max_{\mathbf{W}} \ \sum_{n=1}^{N} R_n^{\mathrm{off}}(\mathbf{w}_n), \qquad (21)$$

which can be equivalently solved by addressing N parallel sub-problems for each $n \in \mathcal{N}$ as

$$(\text{P2.1}) : \max_{\mathbf{w_n}} \ \gamma_n(\mathbf{w}_n) = \frac{\mathbf{w}_n^{\mathrm{H}} \boldsymbol{\Theta}_n \mathbf{w}_n}{\mathbf{w}_n^{\mathrm{H}} \boldsymbol{\Theta}_{-n} \mathbf{w}_n}, \qquad (22)$$

where $\boldsymbol{\Theta}_n = p_n \mathbf{h}_n(\mathbf{h}_n)^{\mathrm{H}}$ and $\boldsymbol{\Theta}_{-n} = \sum_{i=1,i\neq n}^{N} p_i \mathbf{h}_i(\mathbf{h}_i)^{\mathrm{H}} + \sigma^2 \mathbf{I}_M$, with the effective channel $\{\mathbf{h}_n\}_{n\in\mathcal{N}}$ given in (20).

Lemma 2. *It is easy to note that problem (P2.1) in (22) is a generalized eigenvector problem, and its optimal solution $\mathbf{w}_n^*$ should be the eigenvector corresponds to the largest eigenvalue of the matrix $(\boldsymbol{\Theta}_{-n})^{-1} \boldsymbol{\Theta}_n$. Hence, the optimal $\mathbf{w}_n^*$ of problem (P2.1) for $n \in \mathcal{N}$ can be given as*

$$\mathbf{w}_n^* = \mathrm{eigvec}\left\{ \max\left\{ \mathrm{eig}\{(\boldsymbol{\Theta}_{-n})^{-1} \boldsymbol{\Theta}_n\} \right\} \right\}. \qquad (23)$$

We then denote the receive beamforming matrix obtained at the χ-th iteration of the BCD algorithm as $\mathbf{W}_\chi = [\mathbf{w}_1^*, \ldots, \mathbf{w}_N^*]$, which is used in the following subsection.

C. Energy Partition Optimization

Here, the sub-problem (P3) for optimizing the energy partition parameters in $\mathbf{a}$ with given $\boldsymbol{\Phi} = \boldsymbol{\Phi}_\chi$ and $\mathbf{W} = \mathbf{W}_\chi$ is considered, which is given below

$$(\text{P3}) : \max_{\mathbf{a}} \ \sum_{n=1}^{N} \left(R_n^{\mathrm{off}}(\mathbf{a}) + R_n^{\mathrm{loc}}(a_n) \right) \qquad (24a)$$

$$\text{s.t.} \quad a_n \in [0,1], \ \forall n \in \mathcal{N}. \qquad (24b)$$

Note that problem (P3) is non-convex because of the non-concave items $\{R_n^{\mathrm{off}}(\mathbf{a})\}_{n\in\mathcal{N}}$ in the objective function (24a).

Actually, $R_n^{\mathrm{off}}(\mathbf{a})$ for $n \in \mathcal{N}$ can be re-expressed as the difference of two concave functions as follows

$$R_n^{\mathrm{off}}(\mathbf{a}) \triangleq R_{n,1}^{\mathrm{off}}(\mathbf{a}) - R_{n,2}^{\mathrm{off}}(\mathbf{a}_{-n}) = \qquad (25)$$

$$BT \log_2 \left(\sum_{j=1}^{N} a_j \widetilde{E}_j |\mathbf{w}_n^{\mathrm{H}} \mathbf{h}_j|^2 + \sigma^2 ||\mathbf{w}_n^{\mathrm{H}}||^2 \right) -$$

$$BT \log_2 \left(\sum_{i=1,i\neq n}^{N} a_i \widetilde{E}_i |\mathbf{w}_n^{\mathrm{H}} \mathbf{h}_i|^2 + \sigma^2 ||\mathbf{w}_n^{\mathrm{H}}||^2 \right),$$

where $\mathbf{a}_{-n} = [a_1, \ldots, a_{n-1}, a_{n+1}, \ldots, a_N]$.

Then the problem (P3) can also be solved with the DC programming method, where the second concave function in (25), i.e., $R_{n,2}^{\mathrm{off}}(\mathbf{a}_{-n})$, can be substituted by its linear upper bound, so as to obtain a concave approximation of $R_n^{\mathrm{off}}(\mathbf{a})$. Assuming $\mathbf{a}^{(m)}$ is the solution obtained at the m-th ($m = 0, 1, \ldots$) iteration of the DC programming, a linear upper bound of $R_{n,2}^{\mathrm{off}}(\mathbf{a}_{-n})$ at the point $\mathbf{a}^{(m)}$ can be obtained through the first-order Taylor series expansion as

$$R_{n,2}^{\mathrm{off}}(\mathbf{a}_{-n}) \leq \widehat{R}_{n,2}^{\mathrm{off}}(\mathbf{a}_{-n}; \mathbf{a}_{-n}^{(m)}) \qquad (26)$$

$$= R_{n,2}^{\mathrm{off}}(\mathbf{a}_{-n}^{(m)}) + \sum_{i=1,i\neq n}^{N} R_{n,2,i}^{\mathrm{off}\prime}(\mathbf{a}_{-n}^{(m)}) * (a_i - a_i^{(m)}),$$

where $R_{n,2,i}^{\mathrm{off}\prime}(\mathbf{a}_{-n}^{(m)}) = \dfrac{BT}{\ln 2} \dfrac{\widetilde{E}_i |\mathbf{w}_n^{\mathrm{H}} \mathbf{h}_i|^2}{\sum_{j=1,j\neq n}^{N} a_j^{(m)} \widetilde{E}_j |\mathbf{w}_n^{\mathrm{H}} \mathbf{h}_j|^2 + \sigma^2 ||\mathbf{w}_n^{\mathrm{H}}||^2}$ is the first-order derivative of $R_{n,2}^{\mathrm{off}}(\mathbf{a}_{-n})$ w.r.t. a_i at the point $\mathbf{a}_{-n}^{(m)}$. It is easy to note that the equality holds when $\mathbf{a}_{-n} = \mathbf{a}_{-n}^{(m)}$. At the $(m+1)$-th iteration of DC programming, we aim at maximizing the following approximation problem

$$(\text{P3.1}) : \max_{\mathbf{a}} \ \sum_{n=1}^{N} \left(R_{n,1}^{\mathrm{off}}(\mathbf{a}) - \widehat{R}_{n,2}^{\mathrm{off}}(\mathbf{a}_{-n}; \mathbf{a}_{-n}^{(m)}) + R_n^{\mathrm{loc}}(a_n) \right) \qquad (27a)$$

$$\text{s.t.} \quad a_n \in [0,1], \ \forall n \in \mathcal{N}, \qquad (27b)$$

which is a convex problem and can be easily solved by CVX [36]. Through solving problem (P3.1) with CVX, the optimal solution, i.e., $\mathbf{a}^{(m+1)}$, can be finally obtained.

Lemma 3. *The objective function of problem (P3) in (24a) is monotonic increasing w.r.t the iteration index m as*

$$R_n^{\mathrm{off}}(\mathbf{a}^{(m+1)}) + R_n^{\mathrm{loc}}(a_n^{(m+1)}) \qquad (28)$$

$$\overset{(a)}{\geq} R_{n,1}^{\mathrm{off}}(\mathbf{a}^{(m+1)}) - \widehat{R}_{n,2}^{\mathrm{off}}(\mathbf{a}_{-n}^{(m+1)}; \mathbf{a}_{-n}^{(m)}) + R_n^{\mathrm{loc}}(a_n^{(m+1)})$$

$$\overset{(b)}{\geq} R_{n,1}^{\mathrm{off}}(\mathbf{a}^{(m)}) - \widehat{R}_{n,2}^{\mathrm{off}}(\mathbf{a}_{-n}^{(m)}; \mathbf{a}_{-n}^{(m)}) + R_n^{\mathrm{loc}}(a_n^{(m)})$$

$$= R_n^{\mathrm{off}}(\mathbf{a}^{(m)}) + R_n^{\mathrm{loc}}(a_n^{(m)}),$$

where (a) comes from the inequality (26) and (b) holds since $\mathbf{a}^{(m)}$ is a feasible solution while $\mathbf{a}^{(m+1)}$ is the optimal solution of problem (P3.1) in (27). Besides, the objective (24a) is upper-bounded due to the limited energy supply of UEs. In summary, the convergence of the proposed DC programming method for solving problem (P3) in (24) can be guaranteed [34]. We can obtain the final solution of $\mathbf{a}$ at the χ-th iteration of the BCD algorithm when the DC programming converges at the $(m+1)$-th iteration, which is denoted as $\mathbf{a}_\chi = [a_1^{(m+1)}, \ldots, a_N^{(m+1)}]$.

D. Benchmark with Zero Forcing (ZF) Receive Beamforming

An important benchmark scheme of our proposed algorithm is leveraging ZF receive beamforming at the AP, and thus the receive beamforming matrix obtained in Section III-B should be replaced as $\mathbf{W}_\chi = [\mathbf{w}_{1,\mathrm{ZF}}, \ldots, \mathbf{w}_{N,\mathrm{ZF}}] = \mathbf{H}(\mathbf{H}^\mathrm{H}\mathbf{H})^{-1}$ in the χ-th iteration of the proposed BCD algorithm, where $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_N] \in \mathbb{C}^{M \times N}$ is the compact matrix of the UEs' equivalent channels. For the case of $M \geq N$ with independent and identically distributed (i.i.d.) channels, the interference among the UEs can be eliminated, and thus the offloading rate of UE n can be rewritten as

$$R_{n,\mathrm{ZF}}^{\mathrm{off}}(a_n) = BT \log_2\left(1 + \frac{a_n \widetilde{E}_n}{\sigma^2 ||\mathbf{w}_{n,\mathrm{ZF}}^\mathrm{H}||^2}\right), \ \forall n \in \mathcal{N}, \quad (29)$$

which is concave w.r.t. a_n. Then the problem (P3) in (24) is reduced to a convex optimization problem, which can be optimally solved by CVX in a parallel fashion by addressing N sub-problems for $n \in \mathcal{N}$ given below

$$(\mathrm{P3.2}) : \max_{a_n} \ \left(R_{n,\mathrm{ZF}}^{\mathrm{off}}(a_n) + R_n^{\mathrm{loc}}(a_n)\right) \qquad (30\mathrm{a})$$

$$\mathrm{s.t.} \quad a_n \in [0, 1]. \qquad (30\mathrm{b})$$

It is known that the ZF receive beamforming cannot effectively deal with the cases when $M < N$, while our proposed optimal solution in Section III-B can perform well even in these cases, which will be validated in the simulation results.

E. Algorithm, Convergence, and Complexity

The proposed three-step BCD optimization algorithm for solving the original TCTB maximization problem (P0) in (8) is summarized in **Algorithm** 1, through which problem (P0) can be effectively solved with guaranteed convergence [37]. In fact, the convergence can be easily proved based on Lemma 1, 2, and 3, and we can show that the objective function of problem (P0) in (8a) gradually increases with the iteration index χ through updating ϕ, $\mathbf{W}$, and $\mathbf{a}$ iteratively.

The computational complexity of the proposed three-step BCD optimization Algorithm 1 in each iteration mainly lies in the DC programming for solving problem (P1) to design the RIS reflecting coefficients and problem (P3) to optimize the energy partition parameters. Note that the complexity of solving problem (P1.1) in (18) and problem (P3.1) in (27) can be estimated as with the order of $\mathcal{O}(K^6)$ and $\mathcal{O}(N^{3.5})$ according to the complexity of solving convex problems with interior point method [38]. Denote the number of iterations for solving problem (P1) and problem (P3) as L_1 and L_3, respectively, thus the total computational complexity of Algorithm 1 can be further given as $O(L(L_1 K^6 + L_3 N^{3.5}))$, where L represents the total iteration number of the the BCD algorithm. It is easy to observe that the complexity of the proposed algorithm increases dramatically with the number of RIS reflecting elements and the number of UEs.

IV. DEEP LEARNING ARCHITECTURES

The proposed BCD Algorithm 1 provides an effective optimization method for solving the TCTB maximization roblem (P0) in an iterative way. Although the BCD optimization

---

Algorithm 1 Three-Step BCD Optimization Algorithm for Solving the Original TCTB Maximization Problem (P0)

1: **Input** T, N, M, K, B, $\{E_n, C_n, \kappa_n, \mathbf{h}_{\mathrm{d},n}, \mathbf{h}_{\mathrm{r},n}\}_{n \in \mathcal{N}}$, $\mathbf{H}_{\mathrm{AP}}$, and the tolerant thresholds ε, ε_1, ε_3;
2: **Initialize** The iteration index $\chi = 0$ and ϕ_0, $\mathbf{W}_0$, $\mathbf{a}_0$;
3: **Repeat**
4: $\quad \chi = \chi + 1$;
5: $\quad$ **Step 1: Initialize** $l = 0$, $\phi^{(0)} = \phi_{\chi-1}$, $\mathbf{W} = \mathbf{W}_{\chi-1}$, $\mathbf{a} = \mathbf{a}_{\chi-1}$;
6: $\quad$ **Repeat 1**
7: $\quad\quad$ a) Solve problem (P1.1) with CVX to obtain $\mathbf{\Psi}^{(l+1)}$;
8: $\quad\quad$ b) Calculate $R_1^{(l+1)} = \sum_{n=1}^N F_{1,n}(\mathbf{\Psi}^{(l+1)}) - F_{2,n}(\mathbf{\Psi}^{(l+1)})$;
9: $\quad\quad$ c) $l = l + 1$;
10: $\quad$ **End Repeat 1** until convergence, i.e., $|R_1^{(l)} - R_1^{(l-1)}| < \varepsilon_1$ $(l > 1)$, and obtain $\mathbf{\Psi}_\chi = \mathbf{\Psi}^{(l)}$; Then we decompose $\mathbf{\Psi}_\chi = \widetilde{\phi}_\chi \widetilde{\phi}_\chi^\mathrm{H}$ with denoting $\widetilde{\phi}_\chi = [\phi_{\chi,0}, \xi_{\chi,0}]^\mathrm{T}$; Thus, we can obtain $\phi_\chi = \phi_{\chi,0}/\xi_{\chi,0}$ and $\mathbf{\Phi}_\chi = \mathrm{diag}\{\phi_\chi\}$;
11: $\quad$ **Step 2: Initialize** $\phi = \phi_\chi$ and $\mathbf{a} = \mathbf{a}_{\chi-1}$;
12: $\quad$ Obtain $\mathbf{W}_\chi$ according to **Lemma** 2;
13: $\quad$ **Step 3: Initialize** $m = 0$, $\mathbf{a}^{(0)} = \mathbf{a}_{\chi-1}$, $\phi = \phi_\chi$, $\mathbf{W} = \mathbf{W}_\chi$;
14: $\quad$ **Repeat 3**
15: $\quad\quad$ a) Solve problem (P3.1) with CVX to obtain $\mathbf{a}^{(m+1)}$;
$\quad\quad$ b) Calculate the TCTB at the $(m+1)$-th iteration, represented as $R_3^{(m+1)} = \sum_{n=1}^N \left(R_n^{\mathrm{off}}(\mathbf{a}^{(m+1)}) + R_n^{\mathrm{loc}}(a_n^{(m+1)})\right)$;
$\quad\quad$ c) $m = m + 1$;
16: $\quad$ **End Repeat 3** until convergence, i.e., $|R_3^{(m)} - R_3^{(m-1)}| < \varepsilon_3$ $(m > 1)$, and obtain $\mathbf{a}_\chi = \mathbf{a}^{(m)}$;
17: Calculate the TCTB at the χ-th iteration, which is denoted as $R_\chi = \sum_{n=1}^N \left(R_n^{\mathrm{off}}(\mathbf{a}_\chi, \mathbf{w}_{n,\chi}, \phi_\chi) + R_n^{\mathrm{loc}}(a_{n,\chi})\right)$ by substituting $\mathbf{W}_\chi$, $\mathbf{a}_\chi$ and ϕ_χ into the objective function of problem (P0).
18: **End Repeat** until convergence, i.e., $|R_\chi - R_{\chi-1}| < \varepsilon$ $(\chi > 1)$, and obtain the maximum TCTB R_χ with the solution $\mathbf{W}^* = \mathbf{W}_\chi$, $\mathbf{a}^* = \mathbf{a}_\chi$, $\phi^* = \phi_\chi$.

---

algorithm can achieve effective solutions with guaranteed convergence, its high computational complexity may hinder it from being applied in real-time applications, which is a major bottleneck for most of the iterative optimization algorithms proposed in existing works [12–15]. However, the effectiveness, robustness, and computational overhead of online implementations are known as crucial indicators for practical networks. One way to overcome this drawback is leveraging the deep learning methods, not only due to the fact that DNNs are regarded as universal function approximators but also because deep learning is well known as a promising way to achieve effective online implementations [16–18]. Hence, in this section, we explore the potentials of deep learning approaches in obtaining effective solutions of the original problem (P0).

The proposed deep learning methods in this section aim at reducing the computational complexity of the proposed BCD optimization algorithm by effectively emulating this algorithm via supervised learning, so as to facilitate lightweight online implementation of the BCD algorithm with high accuracy and stability. Moreover, the use of DNNs enabled the design of a location-only deep learning approach that reduces overheads for CSI estimation and feedback compared to the BCD algorithm. Through training the constructed DNNs offline with data samples generated from the BCD algorithm, the DNNs are capable of learning the inherent mappings of the algorithm and output effective solutions mimicking

this algorithm. Hence, we can use the trained DNNs to predict the required solutions online with significantly reduced computational complexity/running time. Specifically, we resort to the deep learning approaches to obtain partial solution of problem (P0), including the RIS reflecting coefficients in ϕ and the energy partition parameters in $\mathbf{a}$. Then, we can directly obtain the receive beamforming vectors in $\mathbf{W}$ via Lemma 2 without learning. In this way, we can effectively combine the prior knowledge (Lemma 2) with deep learning to achieve the required solution of the proposed BCD algorithm, with reduced cost for constructing, training, and testing the DNNs.

As mentioned before, the channel links between UEs and the RIS as well as that between the RIS and AP are very likely to be LoS channels, when the location of the RIS is carefully planned. For the multiplexing computation offloading mechanism considered in this paper, the direct links between UEs and the AP play a significant role in providing multi-path diversity gain for wireless communications. Two typical RIS-aided edge computation offloading scenarios in terms of whether LoS direct links exist between UEs and AP are considered here as shown in Fig. 2, where one is without LoS direct links denoted as scenario (a) and the other is with strong LoS direct links denoted as scenario (b). In order to effectively deal with these two scenarios, we construct two deep learning architectures in the following two subsections.
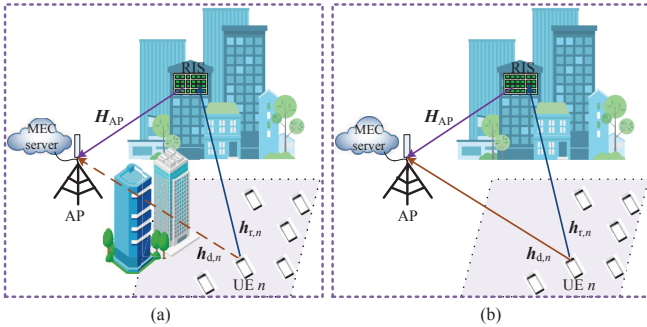

Fig. 2. Two typical RIS-aided edge computation offloading scenarios in terms of whether LoS direct links exist between UEs and the AP: (a) Scenario without LoS direct links between UEs and AP where the LoS direct paths are blocked by objects such as buildings (urban area); (b) Scenario with strong LoS direct links between UEs and AP where no obstruction exists along the direct signal paths (suburb area).

A. CSI-Based Deep Learning Architecture

For scenario (a) without LoS direct links between UEs and AP, a CSI-based deep learning architecture is given, as shown in Fig. 3, to obtain the solutions of $\{\phi, \mathbf{a}, \mathbf{W}\}$. The real and imaginary parts of the channel coefficients $\{\mathbf{h}_{d,n}\}$, $\{\mathbf{h}_{r,n}\}$ and $\mathbf{H}_{AP}$ constitute the input feature of the constructed DNN-CSI, represented by the input vector $\mathbf{x}$ with a dimension of $I = 2(MN+KN+MK)$. In contrast, the normalized angles of the RIS reflecting coefficients ϕ, denoted as $\widetilde{\boldsymbol{\theta}} = \boldsymbol{\theta}/2\pi$, and the energy partition parameters in $\mathbf{a}$ constitute the corresponding output vector $\mathbf{y} = [\widetilde{\theta}_1, \ldots, \widetilde{\theta}_K, a_1, \ldots, a_N]$ with the dimension of $(K + N)$. It is easy to note that all the elements of the output vector are within the range of $[0, 1]$, and thus we can use the sigmoid function, i.e., $\text{Sigmoid}(z) = \frac{1}{1+e^{-z}}$ as the output activation function. With the final output ϕ and $\mathbf{a}$ of the DNN-CSI, the optimal receive beamforming vector for each

UE $n \in \mathcal{N}$, i.e., $\mathbf{w}_n$, can be readily obtained according to Lemma 2 in Section III-B.
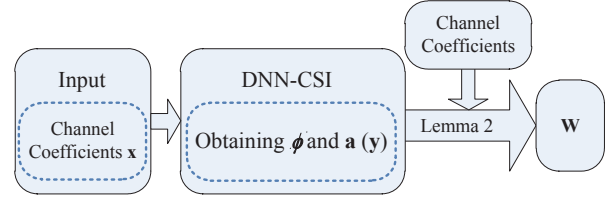

Fig. 3. The architecture for obtaining the solutions of $\{\phi, \mathbf{a}, \mathbf{W}\}$ with the CSI-based DNN-CSI.

TABLE I
LAYOUT OF DNN-CSI FOR OBTAINING ϕ AND $\mathbf{a}$

Layer	Size	Parameter	Activation
Input Layer	I	-	-
Layer1-1 (Dense)	1024	$1024(I+1)$	ELU
Layer1-2 (BN)	1024	4096	-
Layer1-3 (Dropout 0.1)	1024	0	-
Layer2-1 (Dense)	512	524800	ELU
Layer2-2 (BN)	512	2048	-
Layer2-3 (Dropout 0.1)	512	0	-
Layer3-1 (Dense)	256	131328	ELU
Layer3-2 (BN)	256	1024	-
Layer3-3 (Dropout 0.1)	256	0	-
Layer4-1 (Dense)	128	32869	ELU
Layer4-2 (BN)	128	512	-
Layer4-3 (Dropout 0.1)	128	0	-
Layer5-1 (Dense)	128	16512	ELU
Layer5-2 (BN)	128	512	-
Layer5-3 (Dropout 0.05)	128	0	-
Output Layer (Dense)	$K+N$	$(K+N)(128+1)$	sigmoid
Total Trainable Parameters	1,632,288 ($M=8,N=8,K=24$)

Here, we adopt a feedforward DNN with the layout in Table I, which consists of an I-dimensional input layer, 5 normal hidden dense layers (layers 1-1, 2-1, 3-1, 4-1, 5-1), and a $(K + N)$-dimensional output layer, which are the key functional layers of DNN-CSI. There are respectively 1024, 512, 256, 128, 128 neurons for the five hidden layers of the DNN-CSI. The function of exponential linear units (ELU) is leveraged as the activation functions of the hidden layers with

$$\text{ELU}(z) = \begin{cases} z, & \text{if } z > 0, \\ \alpha(\exp(z) - 1), & \text{otherwie}, \ z \leq 0, \end{cases} \quad (31)$$

which has many attractive advantages such as high learning speed, high robustness with zero-centered outputs, etc.[2] It should be noted that we add the Batch-Normalization layers (layers 1-2, 2-2, 3-2, 4-2, 5-2) and Dropout layers (layers 1-3, 2-3, 3-3, 4-3, 5-3) between two normal dense layers to accelerate the training speed, avoid gradients vanishing, as well as prevent overfitting of the DNN. To be specific, this fully connected feedforward DNN-CSI is with 10% of random dropout of neurons for the hidden layer 1 to hidden layer 4 and 5% of random dropout for the hidden layer 5 during each training epoch, so as to avoid overfitting.

B. Location-Only Deep Learning Architecture

For scenario (b) with strong LoS direct links between UEs and AP, the CSI-based deep learning architecture given in the previous subsection is still applicable. Nevertheless, note that

---

[2]We use the default form of ELU function in the Keras platform with α=1.

the channel coefficients as well as the solutions of $\{\phi, \mathbf{a}, \mathbf{W}\}$ are highly related to the locations of UEs, i.e., $\{(x_n, y_n)\}_{n \in \mathcal{N}}$, in this scenario, and thus we may use the UEs' locations as the only input feature of DNNs to obtain $\{\phi, \mathbf{a}, \mathbf{W}\}$. In our considered scenarios, we assume that each UE is installed with an advanced global positioning system (GPS) module for outdoor localization [39] and is capable to apply the Wi-Fi round-trip time technology and standards for indoor localization [40], through which UEs' location information can be obtained with high accuracy.[3]

In order to further use the known relations between the solutions, e.g., Lemma 2 presenting the relationship between $\{\mathbf{W}\}$ with $\{\phi, \mathbf{a}\}$ and the channel coefficients, a location-only deep learning architecture is proposed as shown in Fig. 4. Here, two DNNs are constructed, where DNN-Loc1 aims at calculating the channel mapping between the UEs' locations and the channel coefficients with $2N$-dimensional input feature $\mathbf{z}$ and I-dimensional output feature denoted as $\mathbf{y}_1$ while DNN-Loc2 focuses on obtaining $\{\phi, \mathbf{a}\}$ with input feature $\mathbf{z}$ and $(K+N)$-dimensional output feature denoted as $\mathbf{y}_2$. Then the optimal receive beamforming matrix, i.e., $\mathbf{W}$, can be easily calculated based on Lemma 2 in Section III-B. Note that the complicated pilot channel estimation and feedback can be removed when utilizing the location-only deep learning architecture for online implementation.
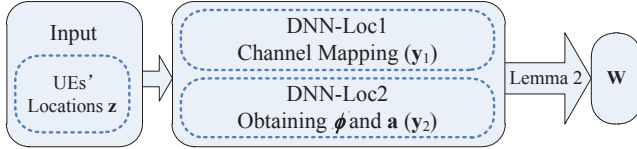

Fig. 4. The architecture for obtaining the solutions of $\{\phi, \mathbf{a}, \mathbf{W}\}$ with the location-only DNN-Loc1 and DNN-Loc2.

TABLE II
LAYOUT OF DNN-LOC1 FOR CHANNEL MAPPING

Layer	Size	Parameter	Activation
Input Layer	$2N$	-	-
Layer1-1 (Dense)	512	512(2N+1)	ELU
Layer1-2 (BN)	512	2048	-
Layer1-3 (Dropout 0.1)	512	0	-
Layer2-1 (Dense)	512	262656	ELU
Layer2-2 (BN)	512	2048	-
Layer2-3 (Dropout 0.1)	512	0	-
Layer3-1 (Dense)	256	131328	ELU
Layer3-2 (BN)	256	1024	-
Layer3-3 (Dropout 0.1)	256	0	-
Layer4-1 (Dense)	128	32869	ELU
Layer4-2 (BN)	128	512	-
Layer4-3 (Dropout 0.1)	128	0	-
Layer5-1 (Dense)	256	33024	ELU
Layer5-2 (BN)	256	1024	-
Layer5-3 (Dropout 0.05)	256	0	-
Output Layer (Dense)	I	I(256+1)	sigmoid
Total Trainable Parameters	702,208 (M=8,N=8,K=24)		

The layout of the feedforward DNN-Loc1 and DNN-Loc2 are given in Table II and III, respectively, where both have

[3]According to the data shown in the website of GPS.gov, for high-end users with dual-frequency receivers and/or augmentation systems, the GPS accuracy can be dramatically boosted, which can enable real-time positioning within a few centimeters and long-term measurements at the millimeter level [39]. It is shown in [40] that an one-meter accuracy indoor localization is available for smart devices through the Wi-Fi round-trip time technology by 2018.

5 normal hidden dense layers. There are respectively 512, 512, 256, 128, 256 neurons for the five hidden layers of the DNN-Loc1, and respectively 512, 256, 128, 64, 32 neurons for the five hidden layers of the DNN-Loc2. Similarly, the layers of Batch-Normalization and Dropout are also utilized for the DNN-Loc1 and DNN-Loc 2 with same dropout police of the DNN-CSI. Here, the sigmoid activation function is not only leveraged at the output layer of the DNN-Loc2 for obtaining $\{\phi, \mathbf{a}\}$ but also at that of the DNN-Loc1 for channel mapping where the output data samples are scaled into the range of $[0, 1]$ with the MinMaxScaler in Tensorflow. In the testing stage, an inverse transformation of MinMaxScaler is used to achieve the required form of the output feature.

TABLE III
LAYOUT OF DNN-LOC2 FOR OBTAINING ϕ AND $\mathbf{a}$

Layer	Size	Parameter	Activation
Input Layer	$2N$	-	-
Layer1-1 (Dense)	512	512(2N+1)	ELU
Layer1-2 (BN)	512	2048	-
Layer1-3 (Dropout 0.1)	512	0	-
Layer2-1 (Dense)	256	131328	ELU
Layer2-2 (BN)	256	1024	-
Layer2-3 (Dropout 0.1)	256	0	-
Layer3-1 (Dense)	128	32896	ELU
Layer3-2 (BN)	128	512	-
Layer3-3 (Dropout 0.1)	128	0	-
Layer4-1 (Dense)	64	8256	ELU
Layer4-2 (BN)	64	256	-
Layer4-3 (Dropout 0.1)	64	0	-
Layer5-1 (Dense)	32	2080	ELU
Layer5-2 (BN)	32	128	-
Layer5-3 (Dropout 0.05)	32	0	-
Output Layer (Dense)	K+N	(K+N)(32+1)	sigmoid
Total Trainable Parameters	186,304 (M=8,N=8,K=24)		

C. Input Feature Uncertainty

In the previous subsection, an ideal scenario is considered where we assume that the input features to the CSI-based and the location-only DNNs, i.e., the input vector $\mathbf{x}$ of CSI in Fig. 3 and the input vector $\mathbf{z}$ of UEs' locations in Fig. 4, are perfectly known. In this case, the constructed DNNs can be trained and tested based on the perfect input information of CSI and UEs' locations. However, the obtained CSI and UEs' locations are usually imperfect in practice due to the deviation of channel estimation and GPS/Wi-Fi localization.

In this section, we focus on a more practical scenario where the input features of CSI and UEs' locations for the CSI-based DNN and the location-only DNNs are corrupted with uncertainty. For the input vector $\mathbf{x}$ of CSI, the corresponding corrupted counterpart is $\hat{\mathbf{x}} = \mathbf{x} + \triangle \mathbf{x}$, where $\triangle \mathbf{x} \sim \mathcal{N}(0, \sigma^2_{\triangle \mathbf{x}})$ following the normal distribution is the random offset of the achieved CSI to the perfect CSI. For the input vector $\mathbf{z}$ of UEs' locations, the corresponding corrupted counterpart is $\hat{\mathbf{z}} = \mathbf{z} + \triangle \mathbf{z}$, where $\triangle \mathbf{z} \sim \mathcal{N}(0, \sigma^2_{\triangle \mathbf{z}})$ is the random offset (in meter) of the achieved UEs' locations to the perfect ones.[4]

In this practical case with uncertain input features, the CSI-based DNN and the location-only DNNs are trained and tested

[4]In the simulation results, the default standard deviation of $\sigma_{\triangle \mathbf{x}}$ is set as 0.001 according to the setting in [41] and the default standard deviation of $\sigma_{\triangle \mathbf{z}}$ is set as 1 based on the localization accuracy of GPS given in GPS.gov [39] and the Wi-Fi round-trip time technology shown in [40].

based on the corrupted input information of CSI and UEs' locations, respectively. The effectiveness and robustness of the two proposed deep learning architectures are validated by comparing their performance in the cases with perfect and imperfect input features, also comparing with the BCD optimization algorithm, which will be shown in Section VI.

V. Implementation and Complexity Reduction of the Deep Learning Approaches

In this section, we indicate the implementation setting of the proposed CSI-based and location-only deep-learning architectures. In addition, the comparison results of the two deep learning methods as well as the proposed BCD optimization algorithm in terms of the average running time are given, which further validates the potentials of the two proposed data-driven approaches in reducing the computational complexity for achieving lightweight online implementations.

The training (including validation) and testing of the constructed DNNs are implemented based on the platforms of Tensorflow and Keras via supervised learning. Also, the adaptive moment estimation (Adam) optimizer is utilized to train the DNNs with adaptive learning rates. We adopt the mean absolute error (MAE) as the loss function for the CSI-based DNN-CSI given in Table I and the location-only DNN-Loc2 given in Table III for obtaining ϕ and $\mathbf{a}$. Through training the weights and bias terms between layers, the input-output mappings of these two DNNs are driven to emulate the inherent mapping of the proposed BCD optimization algorithm. In contrast, the mean square error (MSE) is leveraged as the loss function for the location-only DNN-Loc1 given in Table II for achieving the location-channel mapping. The other parameters relating to training and testing the constructed DNNs are given in the following Table IV.

TABLE IV
PARAMETERS RELATED TO TRAINING AND TESTING

Parameter	Values
Number of training samples	200000
Number of testing samples	10000
Batch size	128
Number of epochs	1000
Initial learning rate	0.001
Validation split	0.2

A. Comparison between Two Deep Learning Approaches

For the CSI-based deep learning architecture, it is required to obtain CSI in advance via pilot channel estimation for time division duplex (TDD) systems. Due to the random characteristics of wireless channels, it is quite difficult to estimate CSI accurately especially considering the pilot contamination. Moreover, the difficulty for wireless channel estimation may dramatically increase and the accuracy may degrade when the adopted subcarriers are with higher frequencies or the APs are with larger set of antennas which lead to more random multi-path fading or larger dimension size of CSI.

In comparison, UEs' location information is less random and its dimension size being independent of the number of APs' antennas is much smaller than that of the corresponding CSI. Hence, it is much easier and more convenient to obtain UEs' location information in practice. In addition, highly accurate location information for UEs in both outdoor and indoor scenarios can be guaranteed thanks to the advanced GPS modules [39] and the Wi-Fi round-trip time technology [40]. In fact, the location-only deep learning architecture provides a promising way to emulate the proposed BCD algorithm for lightweight online implementation, which can remove the complicated pilot channel estimation and feedback prior to wireless communications for task offloading.

Furthermore, in Table V, we present the values of trainable parameters, training time, testing time, and average inference time of the constructed DNN-CSI, DNN-Loc1, DNN-Loc2 for the case with $M = 8$, $N = 8$, $K = 24$ in both cases with perfect and imperfect input features.[5] It is shown that the time overhead for training and testing the constructed DNNs with imperfect input features is slightly larger than that with perfect input features due to the fact that more complicated input-output mappings need to be figured out. Note that the total required training parameters of the two location-only DNNs are considerably less (nearly half) than that of the DNN-CSI as shown in Table I-III. Also, the training, testing, and inference overhead can be significantly reduced by leveraging the location-only data-driven method, which is verified by the much less required training, testing, and average inference time of DNN-Loc1, DNN-Loc2 shown in Table V that are only around 60% of those for DNN-CSI in both cases.[6] Hence, it is of great benefits to leverage the location-only deep learning architecture in situations that it can provide satisfactory inference solutions, such as in the scenario with strong LoS direct links between UEs and AP.

TABLE V
PROCESSING TIME OF THE PROPOSED ALGORITHMS

Parameter	DNN-CSI	DNN-Loc1	DNN-Loc2
Trainable parameters	1,632,288	702,208	186,304
Training samples	$(\mathbf{x}, \mathbf{y})$	$(\mathbf{z}, \mathbf{y}_1)$	$(\mathbf{z}, \mathbf{y}_2)$
Training time	5.7426 h	3.3504 h	1.3979 h
Testing time	0.3883 s	0.2418 s	0.1025 s
Average inference time	38.83 μs	24.18 μs	10.25 μs
Training samples	$(\hat{\mathbf{x}}, \mathbf{y})$	$(\hat{\mathbf{z}}, \mathbf{y}_1)$	$(\hat{\mathbf{z}}, \mathbf{y}_2)$
Training time	6.0345 h	3.5123 h	1.4862 h
Testing time	0.4015 s	0.2568 s	0.1156s
Average inference time	40.15 μs	25.68 μs	11.56 μs
Average BCD Running Time	28.7 s		

B. Complexity Reduction Compared with the BCD Algorithm

As we mentioned in Section IV, the proposed deep learning methods aim at emulating the proposed BCD algorithm with reduced computational complexity, so as to make it possible for lightweight online implementation. In Table V, the average running time of the BCD algorithm for one realization is also given for comparison, i.e., 28.7 s, which is almost 10^6 times to those of the two proposed deep learning methods that the CSI-base and the location-only DNNs only require

---

[5]The training, testing, and inference time in Table V correspond to the processing time by a computer with 64-bit Intel(R) Core(TM) i5-9600KF CPU @3.7GHz and 32 GB RAM, running Python 3.7.7 and Tensorflow 2.1.0. Note that the training and testing time can be further reduced when implemented through more powerful computing servers.

[6]The training, testing, and inference time of the location-only architecture are the maximum of those for DNN-Loc1 and DNN-Loc2 (such as 3.3504 h, 0.2418 s, 24.18 μs for the case with perfect input feature) since these two DNNs can be trained and tested in a parallel way.

Parameter	Symbol	Value
The parameters related to the square serving area	y_{s}, D	20m, 40m
The location of the AP	$(0, y_{\mathrm{AP}}, H_0)$	(0,20,5) m
The location of the RIS	$(x_{\mathrm{R}}, 0, H_{\mathrm{R}})$	(40,0,20) m
The length of the time slot	T	5 seconds
Number of UEs	N	8
Number of AP's antennas	M	8
Number of RIS's reflecting elements	$K = K_y K_z$	24=8 × 3
Energy budgets of UEs	E_n $(n \in \mathcal{N})$	10 J
Required CPU cycles per bit of UEs	C_n $(n \in \mathcal{N})$	200 cycles/bit
The effective switched capacitance of UEs	$\kappa_n (n \in \mathcal{N})$	10^{-28}
The total system bandwidth	B	40 MHz
The noise power	σ^2	−60dBm
The channel power gain at a reference distance of d_0=1 m	L_0	−10dB
The channel attenuation coefficients	α_{d}, α_{r}, α_{AP}	3.5, 2.5, 2
The standard deviation of the offset to the perfect CSI	$\sigma_{\triangle \mathbf{x}}$	0.001
The standard deviation of the offset to the perfect UEs' locations	$\sigma_{\triangle \mathbf{z}}$	1

38.83 μs (40.15 μs) and 24.18 μs (25.68 μs) for the case with perfect (imperfect) input features. This result effectively validates the ability of the proposed deep learning architectures in reducing the computational complexity/running time for providing lightweight online inference solutions.
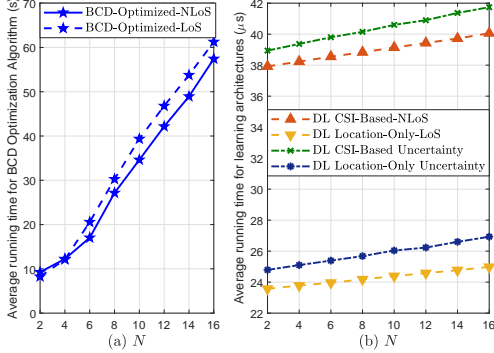

Fig. 5. The average running time for one realization of the proposed solutions versus the number of UEs (N) in the scenarios with NLoS and LoS direct links: (a) Optimization-based solutions, (b) Learning-based solutions without and with input uncertainty.

In Fig. 5, we further show more details of the average running time for one realization of the proposed BCD optimization algorithm, the CSI-based and location-only deep learning architectures versus the number of UEs (N), in the scenarios with NLoS and LoS direct links, respectively. In addition, the corresponding results of the two deep learning methods in the cases with input feature uncertainty are also provided. It is noticeable that the average running time, i.e., the average inference time, of both two deep learning architectures are always quite small in all scenarios (measured in microsecond-μs) and increase slightly with N, especially compared with those of the proposed BCD algorithm (measured in second-s) that increase quite considerably with N. Specifically, the average running time of the CSI-based leaning architecture is nearly millionth and the location-only learning architecture is less than millionth of that required by the corresponding BCD optimization solution, which indicates that lightweight online implementations are available by the proposed two data-driven architectures via periodically training.

VI. SIMULATION RESULTS

In this section, simulation results are given to verify the effectiveness and performance improvement of the proposed BCD optimization algorithm as well as the CSI-based and location-only deep learning architectures. In addition, the effectiveness and robustness of the two proposed deep learning methods to the corrupted input features of CSI and UEs' locations with uncertainty is also validated by simulations.

A three-dimensional (3D) Euclidean coordinate system is adopted to describe the locations of the AP as $(0, y_{\mathrm{AP}}, H)$, the RIS as $(x_{\mathrm{R}}, 0, H_{\mathrm{R}})$ and UE $n \in \mathcal{N}$ as $(x_n, y_n, 0)$, all measured in meters (m) as shown in Fig. 1. The aided RIS is with a uniform rectangular array (URA) of $K = K_y K_z$ reflecting elements, while the M-antenna AP is equipped with a uniform linear array (ULA). We assume that the N ground UEs are randomly distributed in a square serving area of $D \times D$ m^2, with four vertices at horizontal locations of $(x_{\mathrm{s}}, 0)$, $(x_{\mathrm{s}} + D, 0)$, (x_{s}, D), and $(x_{\mathrm{s}} + D, D)$. We consider the Rician fading channel model to account for both the LoS and non-LoS (NLoS) components of all the channels as [42]

$$\mathbf{h}_{\mathrm{r},n} = \sqrt{\frac{\varrho_{\mathrm{r}}}{1 + \varrho_{\mathrm{r}}}} \mathbf{h}_{\mathrm{r},n}^{\mathrm{LoS}} + \sqrt{\frac{1}{1 + \varrho_{\mathrm{r}}}} \mathbf{h}_{\mathrm{r},n}^{\mathrm{NLoS}}, \ \forall n \in \mathcal{N}, \quad (32)$$

$$\mathbf{H}_{\mathrm{AP}} = \sqrt{\frac{\varrho_{\mathrm{AP}}}{1 + \varrho_{\mathrm{AP}}}} \mathbf{H}_{\mathrm{AP}}^{\mathrm{LoS}} + \sqrt{\frac{1}{1 + \varrho_{\mathrm{AP}}}} \mathbf{H}_{\mathrm{AP}}^{\mathrm{NLoS}}, \quad (33)$$

$$\mathbf{h}_{\mathrm{d},n} = \sqrt{\frac{\varrho_{\mathrm{d}}}{1 + \varrho_{\mathrm{d}}}} \mathbf{h}_{\mathrm{d},n}^{\mathrm{LoS}} + \sqrt{\frac{1}{1 + \varrho_{\mathrm{d}}}} \mathbf{h}_{\mathrm{d},n}^{\mathrm{NLoS}}, \ \forall n \in \mathcal{N}, \quad (34)$$

where ϱ_{r}, ϱ_{AP}, ϱ_{d} indicate the corresponding Rician factors. Without loss of generality, we denote $\zeta_{\mathrm{r}} = \frac{\varrho_{\mathrm{r}}}{1 + \varrho_{\mathrm{r}}}$, $\zeta_{\mathrm{AP}} = \frac{\varrho_{\mathrm{AP}}}{1 + \varrho_{\mathrm{AP}}}$, $\zeta_{\mathrm{d}} = \frac{\varrho_{\mathrm{d}}}{1 + \varrho_{\mathrm{d}}}$ as the Rician parameters related to the LoS components which are used to generate the channels in the simulations. Assuming that a half-wavelength spacing is assumed among adjacent elements/antennas at the RIS and AP, the LoS components modeled in the angular domain are then given as [42, 43]

$$\mathbf{h}_{\mathrm{r},n}^{\mathrm{LoS}} = \sqrt{L_{\mathrm{r},n}} \mathbf{e}_{\mathrm{r},n}^{\mathrm{r}}(\beta_{\mathrm{r},n}^{\mathrm{r}}, \gamma_{\mathrm{r},n}^{\mathrm{r}}), \ \forall n, \quad (35)$$

$$\mathbf{H}_{\mathrm{AP}}^{\mathrm{LoS}} = \sqrt{L_{\mathrm{AP}}} \mathbf{e}_{\mathrm{AP}}^{\mathrm{r}}(\beta_{\mathrm{AP}}^{\mathrm{r}})(\mathbf{e}_{\mathrm{R}}^{\mathrm{t}}(\beta_{\mathrm{R}}^{\mathrm{t}}, \gamma_{\mathrm{R}}^{\mathrm{t}}))^{\mathrm{H}}, \quad (36)$$

$$\mathbf{h}_{\mathrm{d},n}^{\mathrm{LoS}} = \sqrt{L_{\mathrm{d},n}} \mathbf{e}_{\mathrm{d},n}^{\mathrm{r}}(\beta_{\mathrm{d},n}^{\mathrm{r}}), \ \forall n, \quad (37)$$

where $\mathbf{e}_{\mathrm{r},n}^{\mathrm{r}}(\beta_{\mathrm{r},n}^{\mathrm{r}},\gamma_{\mathrm{r},n}^{\mathrm{r}}) \in \mathbb{C}^{K\times 1} = \mathbf{e}_{\mathrm{r},n,y}^{\mathrm{r}}(\beta_{\mathrm{r},n}^{\mathrm{r}},\gamma_{\mathrm{r},n}^{\mathrm{r}}) \otimes \mathbf{e}_{\mathrm{r},n,z}^{\mathrm{r}}(\beta_{\mathrm{r},n}^{\mathrm{r}},\gamma_{\mathrm{r},n}^{\mathrm{r}})$ with $\mathbf{e}_{\mathrm{r},n,y}^{\mathrm{r}}(\beta_{\mathrm{r},n}^{\mathrm{r}},\gamma_{\mathrm{r},n}^{\mathrm{r}}) = \{\exp(j\pi(k_y-1)\sin\beta_{\mathrm{r},n}^{\mathrm{r}}\sin\gamma_{\mathrm{r},n}^{\mathrm{r}})\}_{k_y=1}^{K_y} \in \mathbb{C}^{K_y\times 1}$ and $\mathbf{e}_{\mathrm{r},n,z}^{\mathrm{r}}(\beta_{\mathrm{r},n}^{\mathrm{r}},\gamma_{\mathrm{r},n}^{\mathrm{r}}) = \{\exp(j\pi(k_z-1)\cos\beta_{\mathrm{r},n}^{\mathrm{r}}\sin\gamma_{\mathrm{r},n}^{\mathrm{r}})\}_{k_z=1}^{K_z} \in \mathbb{C}^{K_z\times 1}, \mathbf{e}_{\mathrm{AP}}^{\mathrm{r}}(\beta_{\mathrm{AP}}^{\mathrm{r}}) = \{\exp(j\pi(m-1)\sin\beta_{\mathrm{AP}}^{\mathrm{r}})\}_{m=1}^{M} \in \mathbb{C}^{M\times 1}$ and $\mathbf{e}_{\mathrm{d},n}^{\mathrm{r}}(\beta_{\mathrm{d},n}^{\mathrm{r}}) = \{\exp(j\pi(m-1)\sin\beta_{\mathrm{d},n}^{\mathrm{r}})\}_{m=1}^{M} \in \mathbb{C}^{M\times 1}$ are the receive array steering vectors with the effective angles of arrival (AOAs). Also, $\mathbf{e}_{\mathrm{R}}^{\mathrm{t}}(\beta_{\mathrm{R}}^{\mathrm{t}},\gamma_{\mathrm{R}}^{\mathrm{t}}) \in \mathbb{C}^{K\times 1} = \mathbf{e}_{\mathrm{R},y}^{\mathrm{t}}(\beta_{\mathrm{R}}^{\mathrm{t}},\gamma_{\mathrm{R}}^{\mathrm{t}}) \otimes \mathbf{e}_{\mathrm{R},z}^{\mathrm{t}}(\beta_{\mathrm{R}}^{\mathrm{t}},\gamma_{\mathrm{R}}^{\mathrm{t}})$ is the transmit array steering vector with the effective angles of departure (AOD), where $\mathbf{e}_{\mathrm{R},y}^{\mathrm{t}}(\beta_{\mathrm{R}}^{\mathrm{t}},\gamma_{\mathrm{R}}^{\mathrm{t}}) = \{\exp(j\pi(k_y-1)\sin\beta_{\mathrm{R}}^{\mathrm{t}}\sin\gamma_{\mathrm{R}}^{\mathrm{t}})\}_{k_y=1}^{K_y} \in \mathbb{C}^{K_y\times 1}$ and $\mathbf{e}_{\mathrm{R},z}^{\mathrm{t}}(\beta_{\mathrm{R}}^{\mathrm{t}},\gamma_{\mathrm{R}}^{\mathrm{t}}) = \{\exp(j\pi(k_z-1)\cos\beta_{\mathrm{R}}^{\mathrm{t}}\sin\gamma_{\mathrm{R}}^{\mathrm{t}})\}_{k_z=1}^{K_z} \in \mathbb{C}^{K_z\times 1}$. Here, β and γ respectively represent the elevation and azimuth of AOA or AOD.

$L_{\mathrm{r},n}$, L_{AP} and $L_{\mathrm{d},n}$ in (35)-(37) model the distance-dependent path loss of the corresponding channels. Suppose that each element of the RIS has a 3 dBi gain due to the fact that only the front half-space reflects signals [44], while each antenna of the AP has an isotropic radiation pattern with 0 dBi antenna gain. Then we have $L_{\mathrm{r},n} = 10^{0.3}L_0(d_{\mathrm{r},n}/d_0)^{-\alpha_{\mathrm{r}}}$, $L_{\mathrm{AP}} = 10^{0.3}L_0(d_{\mathrm{AP}}/d_0)^{-\alpha_{\mathrm{AP}}}$ and $L_{\mathrm{d},n} = L_0(d_{\mathrm{d},n}/d_0)^{-\alpha_{\mathrm{d}}}$, where $d_{\mathrm{r},n}$, d_{AP}, $d_{\mathrm{d},n}$ are the corresponding Euclidean distances between the transceivers, L_0 is the average constant path loss for all the channels at the reference distance of d_0, and α_{r}, α_{AP}, α_{d} are the channel attenuation coefficients.

In addition, the NLoS components of the channels in (32)-(34) are modeled as the Rayleigh fading combining with the distance-dependent path loss as follows

$$\mathbf{h}_{\mathrm{r},n}^{\mathrm{NLoS}} = \sqrt{L_{\mathrm{r},n}}\boldsymbol{\eta}_{\mathrm{r},n} = \sqrt{10^{0.3}L_0(d_{\mathrm{r},n}/d_0)^{-\alpha_{\mathrm{r}}}}\boldsymbol{\eta}_{\mathrm{r},n}, \quad (38)$$

$$\mathbf{H}_{\mathrm{AP}}^{\mathrm{NLoS}} = \sqrt{L_{\mathrm{AP}}}\boldsymbol{\Gamma}_{\mathrm{AP}} = \sqrt{10^{0.3}L_0(d_{\mathrm{AP}}/d_0)^{-\alpha_{\mathrm{AP}}}}\boldsymbol{\Gamma}_{\mathrm{AP}}, \quad (39)$$

$$\mathbf{h}_{\mathrm{d},n}^{\mathrm{NLoS}} = \sqrt{L_{\mathrm{d},n}}\boldsymbol{\eta}_{\mathrm{d},n} = \sqrt{L_0(d_{\mathrm{d},n}/d_0)^{-\alpha_{\mathrm{d}}}}\boldsymbol{\eta}_{\mathrm{d},n}, \quad (40)$$

where $\boldsymbol{\eta}_{\mathrm{r},n} \sim \mathcal{CN}(0,\mathbf{I}_K)$, $\boldsymbol{\Gamma}_{\mathrm{AP}} \sim \mathcal{CN}(0,\mathbf{I}_M)$, and $\boldsymbol{\eta}_{\mathrm{d},n} \sim \mathcal{CN}(0,\mathbf{I}_M)$ denote the corresponding Rayleigh fading coefficients. In the following simulation results, we assume that the LoS channels between the RIS and the AP as well as UEs are achieved by deploying the RIS at a desirable location, and thus $\zeta_{\mathrm{r}} = \zeta_{\mathrm{AP}} = 1$. The other basic simulation parameters are listed in Table VI unless specified otherwise.

A. Results in the Scenario without LoS Direct Links

In this subsection, the LoS paths of UEs' direct links are blocked as shown in the scenario (a) of Fig. 2, which is quite common in practical communications especially in central building districts of urban areas. Hence, we set $\zeta_{\mathrm{d}} = 0$ in this subsection. Numerical results for the proposed optimization solution ('BCD-Optimized Solution'), the CSI-based deep learning solution ('DL CSI-Based') as well as its counterpart with input uncertainty ('DL CSI-Based Uncertainty') are presented in comparison with three traditional benchmarks, where the 'Direct Offloading-No RIS' scheme corresponds to the case without deploying RIS, the 'ZF Receive Beamforming' scheme considers the ZF beamforming for detecting UEs' signals as in Section III-D, and the 'Equal Energy Allocation'

scheme is operated by equally allocating the UEs' energy budgets for local computing and computation offloading.[7]

In Fig. 6, we first show the TCTB of all the considered schemes w.r.t. the UEs' uniform energy budget, i.e., $E = E_n$ for $n \in \mathcal{N}$. From this figure, we can observe that the TCTB curves of all the schemes increase with E, which coincides with the intuition that more computation task-input bits can be completed if the UEs are endowed with more energy. It is clear to see that significant performance improvement can be achieved by the proposed BCD-Optimized Solution, verifying the great benefits of deploying the aided RIS, also jointly optimizing the RIS coefficients, the receive beamforming and the UEs' energy allocation. It is confirmed that the proposed BCD algorithm provides 26% improvement in TCTB over the benchmark of direct offloading without the assistance of RIS. More importantly, the CSI-based deep learning method can achieve a performance very close to the proposed optimization solution no matter with perfect or imperfect input feature of CSI, which clearly demonstrates that the CSI-based deep learning architecture proposed in Section IV-A can effectively emulate the proposed BCD optimization algorithm, with a much reduced online complexity and high robustness.
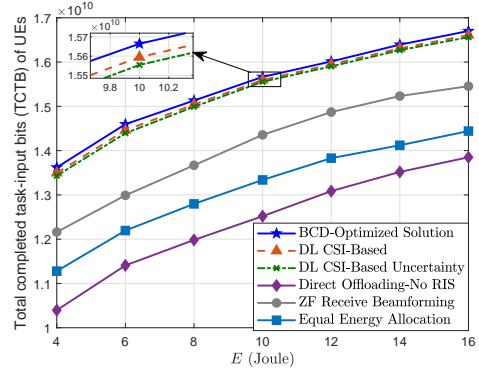

Fig. 6. The TCTB of UEs versus the UEs' uniform energy budget $E = E_n$ for $n \in \mathcal{N}$.
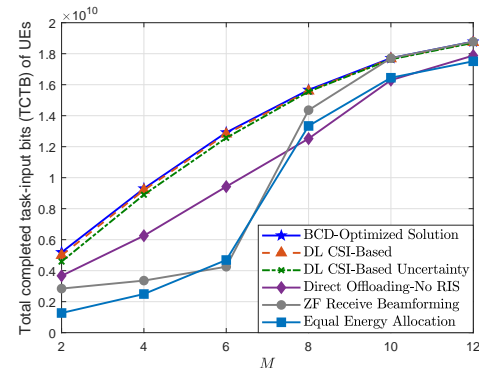

Fig. 7. The TCTB of UEs versus the number of AP's antennas M with $N = 8$.

---

[7]Note that the learning-based works mentioned in the Introduction section either focus on traditional MEC or RIS-aided downlink architectures, and none of them consider the RIS-aided MEC systems. Even though some optimization schemes of RIS-aided MEC systems were given in [12–15], different performance metrics or scenarios were considered as discussed in the Introduction. Hence, these works and our current work are not comparable.

The performance in terms of TCTB versus the number of the AP's antennas is presented in Fig. 7. The effectiveness, robustness, and generalizability of the proposed CSI-Based deep learning architecture is further validated by the results that its performance can always approach that of the BCD-Optimized Solution in both scenarios with and without input uncertainty no matter how many antennas are installed at the AP. Although we can see that all the curves of TCTB increase as M grows, it is obvious that the performance of proposed optimization solution and the DL CSI-Based schemes are far more superior and stable than that of the other baseline solutions especially in the situations of $M < N$. When the AP has to serve more UEs than its installed antennas, the performance gap between the DL CSI-Based with and without input uncertainty is slightly larger, while the performances of the schemes with ZF Receive Beamforming and Equal Energy Allocation degrade dramatically. This is due to the fact that the ZF receive beamforming is incapable of separating out the signal streams when they are more than the number of receive antennas. Also, in these situations, the interference management through designing the UEs' energy allocation plays a significant role in guaranteeing the system performance.

In Fig. 8, we study the effects of the number of UEs, i.e., N, on the system performance of TCTB. Here, the effectiveness of the CSI-based deep learning architecture is further verified in the scenarios with different number of users considering both perfect and imperfect input CSI, which also demonstrates its robustness and the generalizability. Similar results can be observed as from Fig. 7 that our proposed BCD-Optimized Solution as well as the CSI-based deep learning schemes with and without input uncertainty have strong robustness in dealing with the cases when serving more UEs than the number of the AP's antennas. These cases are particularly relevant to massive connectivity scenarios. Instead of degrading the performance like the benchmarks, our proposed solutions are able to provide even better performance as N becoming larger than M through effectively designing the receive bearmforming vectors and the UE's energy allocation.
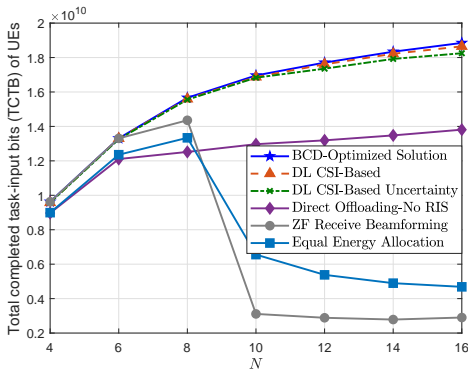

Fig. 8. The TCTB of UEs versus the number of UEs N with $M = 8$.

B. Results in the Scenario with Strong LoS Direct Links

In this subsection, we focus on implementing the mentioned schemes in the scenario where strong LoS direct links exist for UEs in the considered serving area, which is exactly the scenario (b) of Fig. 2. This scenario is practically relevant

when considering the suburb districts, and we set $\zeta_d = 1$ in the following simulation results. In this scenario, the location-only deep learning architecture ('DL Location-Only') can be leveraged to mimic the mapping of the proposed BCD algorithm. In addition, the performance of its counterpart solution with uncertain input UEs' locations denoted as 'DL Location-Only Uncertainty' is also given in this subsection.
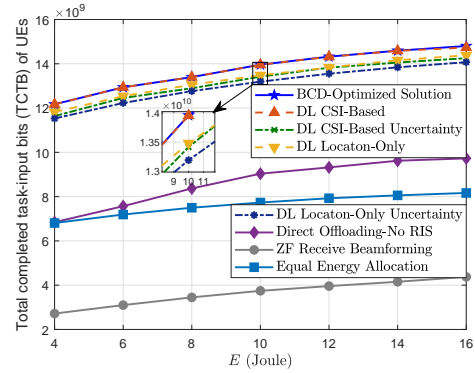

Fig. 9. The TCTB of UEs versus the UEs' uniform energy budget $E = E_n$ for $n \in \mathcal{N}$.

We first show the TCTB performance of all the considered schemes versus the UEs' uniform energy budget E in Fig. 9. Obviously, both the CSI-based and the location-only deep learning methods can achieve excellent system performance, in both scenarios with and without input uncertainty. Even though the DL Location-Only solution is slightly worse than the DL CSI-Based solution which is almost the same as the BCD-Optimized Solution, it is far more superior than the other benchmarks. It is noticeable that the TCTB gap between the DL CSI-Based and DL Location-Only schemes becomes slightly smaller in the scenarios with uncertain input features. More importantly, the UEs' locations are quite easier to obtain compared with the related CSI, which makes it more flexible to achieve online implementation through the location-only deep learning architecture in both scenarios with perfect and imperfect input features. Also from this figure, we can clearly see that the ZF Receive Beamforming scheme is even worse than the scheme without RIS and the scheme of Equal Energy Allocation. The reason behind this is that in the considered scenario with LoS direct links, the effective channels of different UEs may be highly correlated, and it is almost impossible to distinguish different UEs' data streams through ZF receive beamforming especially in the cases with $M \leq N$. This phenomenon further indicates the importance of effectively designing the AP's receive beamforming and managing the UEs' energy budgets.

Fig. 10 depicts the TCTB curves versus the number of the AP's antennas, i.e., M. Clearly, the results in this figure further demonstrate the effectiveness, the robustness and the generalizability of the CSI-based DNN architecture in different scenarios combined with the results in the previous subsection and the location-only DNN architecture in situations where AP is installed with different number of antennas. If only uncertain input features are available, we can observe that the DL Location-only scheme can almost approach the DL CSI-Based when $M < N$. Similar to the results in Fig. 9,
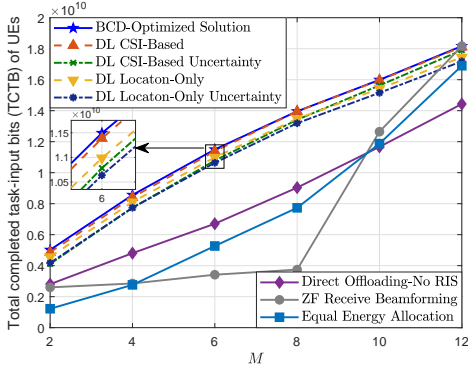
Fig. 10. The TCTB of UEs versus the number of AP's antennas M with $N = 8$.

the performance of the ZF Receive Beamforming scheme is unacceptable when $M \leq N$. When M is greater than N, like $M = 12$, the disadvantages of the scheme without RIS becomes more obvious since all the other schemes can achieve much better performance with the assistance of the RIS.
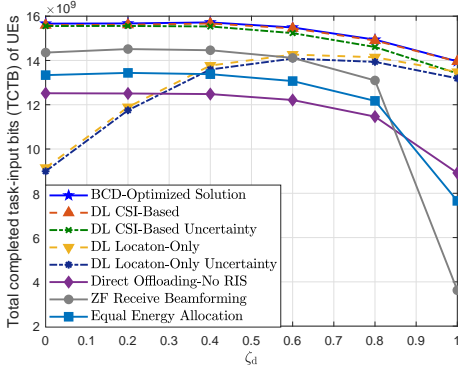

Fig. 11. The TCTB of UEs versus the Rician channel parameter ζ_{d}.

In Fig. 11, we show the influence of the Rician channel parameter related to the LoS components of the direct links, i.e, ζ_{d}. It can be seen that the performance of the DL CSI-Based scheme can always achieve satisfactory performance very close to the proposed BCD-Optimized Solution if perfect CSI is available, no matter the UEs' direct links are highly faded without LoS components as in $\zeta_{\mathrm{d}} = 0$, or with LoS components as $\zeta_{\mathrm{d}} > 0$ and even $\zeta_{\mathrm{d}} = 1$. This result indicates that the perfect input feature of CSI is capable of capturing sufficient information in emulating the proposed BCD optimized solution. In contrast, the DL Location-Only scheme can achieve good performance when ζ_{d} is close to 1, but the performance degrades as ζ_{d} decreases. This is reasonable since the optimization solution becomes less relevant to the UE's locations as ζ_{d} becomes smaller where the small-scale Rayleigh fading accounts more. Compared with the DL CSI-Based scheme, the performance of DL Location-Only solution is more stable and degrades slighter when $\zeta_{\mathrm{d}} = 1$ with uncertain input feature. Fig. 11 verifies that the location-only DNN architecture is more suitable to the scenarios where strong LoS direct links for UEs are present, just coinciding with our original intention as in Section IV-B. Interestingly, we can see that the performance of the schemes except for the two

DL Location-Only solutions degrades as ζ_{d} increases, which highlights that the channel fading is beneficial to wireless communications when considering the multiplexing computation offloading since channel fading provides an additional degree of freedom for avoiding channel correlation [42]. While for the case of $\zeta_{\mathrm{d}} = 1$, the performance becomes worse since the UEs' channels are highly correlated and no fading can be used to achieve the additional degree of freedom gain.
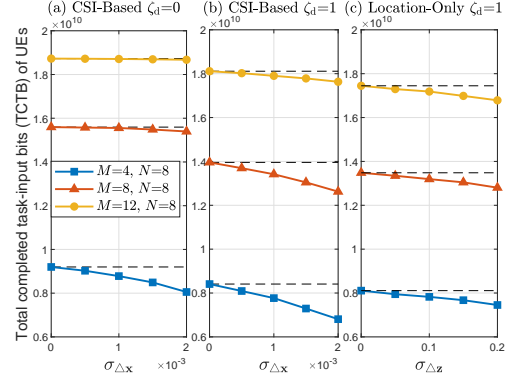

Fig. 12. The TCTB of UEs versus the Input features' uncertainty level.

Fig. 12 presents the TCTB performance of the two deep learning architectures versus the uncertainty levels of the CSI and UEs' locations, respectively represented by $\sigma_{\triangle \mathbf{x}}$ and $\sigma_{\triangle \mathbf{z}}$. Case (a) and (b) show the performance of the CSI-based learning solution in the scenarios with NLoS and LoS direct links, respectively, while case (c) demonstrates that of the location-only learning method in the scenario with LoS direct links. From these three cases, we can observe that the performance degradation is much less when $M \geq N$, which is quite obvious for the CSI-based learning method in (a) and (b). In addition, the CSI-based learning solution degrades more serious in the scenario with LoS direct links in (b) compared with the NLoS case in (a), due to the fact that the DNN-CSI in LoS scenario (b) without channel fading is more sensitive to the input uncertainty in CSI. Based on (b) and (c), it is noticeable that the location-only solution is less sensitive to the uncertain input feature in the scenario with LoS direct links, which further indicates its higher stability and robustness for fulfilling lightweight online implementation in this scenario.

VII. CONCLUSION

In this paper, a RIS-aided MEC architecture with multiplexing computation offloading has been investigated, where the RIS constructively reflects the UEs' offloaded input-data-bearing signals to improve the UEs' computation efficiency. During a given time slot, the TCTB of all the UEs with limited energy budgets is maximized by jointly optimizing the RIS reflecting coefficients, the receiving beamforming vectors and UEs' energy partition strategies for local computing and computation offloading. A three-step BCD optimization algorithm is proposed to solve the formulated non-convex TCTB maximization problem iteratively with guaranteed convergence. In addition, two deep learning architectures based on CSI and the UEs' locations are constructed to mimic the mapping of the BCD algorithm with a considerable complexity reduction. The

simulation results have confirmed that significant performance improvement can be achieved by leveraging the proposed BCD algorithm comparing with some existing schemes. For both scenarios with perfect and imperfect input features, the CSI-based learning architecture can always approach the performance of the BCD algorithm, while the more practical location-only learning architecture can provide satisfactory and more robust performance when strong LoS direct links exist between UEs and AP.

REFERENCES

[1] X. Hu, C. Masouros, and K. Wong, "Removing channel estimation by location-only based deep learning for RIS aided mobile edge computing," in *proc. IEEE Inter. Conf. Commun. (ICC)*, Virtual/Montreal, Canada, June 2021, pp. 1–6.

[2] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, fourthquarter 2017.

[3] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Signal Inf. Process. Netw.*, vol. 1, no. 2, pp. 89–103, Jun. 2015.

[4] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

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

[6] X. Hu, K. Wong, and K. Yang, "Wireless powered cooperation-assisted mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 17, no. 4, pp. 2375–2388, Apr. 2018.

[7] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 856–868, Jan. 2019.

[8] X. Hu, L. Wang, K. Wong, M. Tao, Y. Zhang, and Z. Zheng, "Edge and central cloud computing: A perfect pairing for high energy efficiency and low-latency," *IEEE Trans. Wireless Commun.*, vol. 19, no. 2, pp. 1070–1083, 2020.

[9] E. Basar, M. Di Renzo, J. De Rosny, M. Debbah, M. Alouini, and R. Zhang, "Wireless communications through reconfigurable intelligent surfaces," *IEEE Access*, vol. 7, pp. 116 753–116 773, 2019.

[10] Q. Wu and R. Zhang, "Towards smart and reconfigurable environment: Intelligent reflecting surface aided wireless network," *IEEE Commun. Mag.*, vol. 58, no. 1, pp. 106–112, 2020.

[11] C. Huang, A. Zappone, G. C. Alexandropoulos, M. Debbah, and C. Yuen, "Reconfigurable intelligent surfaces for energy efficiency in wireless communication," *IEEE Trans. Wireless Commun.*, vol. 18, no. 8, pp. 4157–4170, 2019.

[12] T. Bai, C. Pan, Y. Deng, M. Elkashlan, A. Nallanathan, and L. Hanzo, "Latency minimization for intelligent reflecting surface aided mobile edge computing," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 11, pp. 2666–2682, 2020.

[13] S. Hua, Y. Zhou, K. Yang, and Y. Shi, "Reconfigurable intelligent surface for green edge inference," *arXiv preprint arXiv:1912.00820*, 2019.

[14] Y. Liu, J. Zhao, Z. Xiong, D. Niyato, Y. Chau, C. Pan, and B. Huang, "Intelligent reflecting surface meets mobile edge computing: Enhancing wireless communications for computation offloading," *arXiv preprint arXiv:2001.07449*, 2020.

[15] T. Bai, C. Pan, H. Ren, Y. Deng, M. Elkashlan, and A. Nallanathan, "Resource allocation for intelligent reflecting surface aided wireless powered mobile edge computing in OFDM systems," *arXiv preprint arXiv:2003.05511*, 2020.

[16] K. Hornik, M. Stinchcombe, H. White *et al.*, "Multilayer feedforward networks are universal approximators." *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[17] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[18] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, 2017.

[19] J. Chen and X. Ran, "Deep learning with edge computing: A review," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1655–1674, 2019.

[20] L. Huang, X. Feng, A. Feng, Y. Huang, and L. P. Qian, "Distributed deep learning-based offloading for mobile edge computing networks," *Springer: Mobile Networks and Applications*, pp. 1–8, 2018.

[21] J. Wang, L. Zhao, J. Liu, and N. Kato, "Smart resource allocation for mobile edge computing: A deep reinforcement learning approach," *IEEE Trans. Emerg. Topics Comp.*, pp. 1–1, 2019.

[22] L. Huang, S. Bi, and Y. J. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Trans. Mobile Comp.*, pp. 1–1, 2019.

[23] H. Yang, Z. Xiong, J. Zhao, D. Niyato, L. Xiao, and Q. Wu, "Deep reinforcement learning based intelligent reflecting surface for secure wireless communications," pp. 1–1, 2020.

[24] C. Huang, R. Mo, and C. Yuen, "Reconfigurable intelligent surface assisted multiuser MISO systems exploiting deep reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 8, pp. 1839–1850, 2020.

[25] A. M. Elbir, A. Papazafeiropoulos, P. Kourtessis, and S. Chatzinotas, "Deep channel learning for large intelligent surfaces aided mm-wave massive MIMO systems," *IEEE Wireless Commun. Lett.*, vol. 9, no. 9, pp. 1447–1451, 2020.

[26] S. Bi and Y. J. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 4177–4190, 2018.

[27] F. Zhou and R. Q. Hu, "Computation efficiency maximization in wireless-powered mobile edge computing networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3170–3184, 2020.

[28] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian, "Computation rate maximization in UAV-enabled wireless-powered mobile-edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 1927–1941, Sep. 2018.

[29] X. Hu, K. K. Wong, and Y. Zhang, "Wireless-powered edge computing with cooperative UAV: Task, time scheduling and trajectory design," *IEEE Trans. Wireless Commun.*, vol. 19, no. 12, pp. 8083–8098, 2020.

[30] X. Hu, K. Wong, K. Yang, and Z. Zheng, "UAV-assisted relaying and edge computing: Scheduling and trajectory optimization," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4738–4752, Oct. 2019.

[31] Q. Wu and R. Zhang, "Intelligent reflecting surface enhanced wireless network via joint active and passive beamforming," *IEEE Trans. Wireless Commun.*, vol. 18, no. 11, pp. 5394–5409, 2019.

[32] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[33] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Trans. Wireless Commun.*, vol. 12, no. 9, pp. 4569–4581, Sep. 2013.

[34] P. D. Tao *et al.*, "The DC (difference of convex functions) programming and DCA revisited with DC models of real world nonconvex optimization problems," *Annals of operations research*, vol. 133, no. 1-4, pp. 23–46, 2005.

[35] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via over-the-air computation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 2022–2035, 2020.

[36] M. Grant, S. Boyd, and Y. Ye, "CVX: Matlab software for disciplined convex programming," 2008.

[37] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[38] K. Wang, A. M. So, T. Chang, W. Ma, and C. Chi, "Outage constrained robust transmit optimization for multiuser miso downlinks: Tractable approximations by conic optimization," *IEEE Trans. Signal Process.*, vol. 62, no. 21, pp. 5690–5705, 2014.

[39] U.S. Air Force, "Official U.S. government information about the Global Positioning System (GPS) and related topics: GPS Accuracy," GPS.GOV, 2020, https://www.gps.gov/systems/gps/performance/accuracy/.

[40] F. van Diggelen, R. Want, and W. Wang, "How to achieve 1- meter accuracy in android," *Android Location, Google, 2018, https://www.gpsworld.com/how-to-achieve-1-meter-accuracy-in-android/*.

[41] F. Sohrabi, H. V. Cheng, and W. Yu, "Robust symbol-level precoding via autoencoder-based deep learning," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 8951–8955.

[42] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. Cambridge university press, 2005.

[43] L. Yuan, R. Jiang, and Y. Chen, "Gain and phase autocalibration of large uniform rectangular arrays for underwater 3-d sonar imaging systems," *IEEE Journal of Oceanic Engineering*, vol. 39, no. 3, pp. 458–471, 2014.

[44] Q. Wu and R. Zhang, "Joint active and passive beamforming optimization for intelligent reflecting surface assisted swipt under qos constraints," *arXiv preprint arXiv:1910.06220*, 2019.