Unsupervised Deep Learning for Binary Offloading in Mobile Edge Computation Network

Xue Chen
Central China Normal University
Hongbo Xu
Central China Normal University
Guoping Zhang (gpzhang@mail.ccnu.edu.cn)
Central China Normal University
Yun Chen
Central China Normal University
Ruijie Li
Central China Normal University

Research Article

Keywords: MEC, Binary offloading decision, Unsupervised deep learning, Auxiliary network

Posted Date: April 14th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-372831/v1

Version of Record: A version of this preprint was published at Wireless Personal Communications on December 2nd, 2021. See the published version at https://doi.org/10.1007/s11277-021-09433-9.

Noname manuscript No.
(will be inserted by the editor)

Unsupervised Deep Learning for Binary Offloading in Mobile Edge Computation Network

Xue Chen · Hongbo Xu · Guoping Zhang(✉) · Yun Chen · Ruijie Li

Received: date / Accepted: date

Abstract Mobile edge computation (MEC) is a potential technology to reduce the energy consumption and task execution delay for tackling computation-intensive tasks on mobile device (MD). The resource allocation of MEC is an optimization problem, however, the existing large amount of computation may hinder its practical application. In this work, we propose a multiuser MEC framework based on unsupervised deep learning (DL) to reduce energy consumption and computation by offloading tasks to edge servers. The binary offloading decision and resource allocation are jointly optimized to minimize energy consumption of MDs under latency constraint and transmit power constraint. This joint optimization problem is a mixed integer nonconvex problem which result in the gradient vanishing problem in backpropagation. To address this, we propose a novel binary computation offloading scheme (BCOS), in which a deep neural network (DNN) with an auxiliary network is designed. By using the auxiliary network as a teacher network, the student network can obtain the lossless gradient information in joint training phase. As a result, the sub-optimal solution of the optimization problem can be acquired by the learning-based BCOS. Simulation results demonstrate that the BCOS is effective to solve the binary offloading problem by the trained network with low complexity.

Keywords MEC · Binary offloading decision · Unsupervised deep learning · Auxiliary network

✉ Guoping Zhang
E-mail: gpzhang@mail.ccnu.edu.cn
Department of Electronics and Information Engineering, Central China Normal University, Wuhan 430000, China

1 Introduction

With the dramatic growth of Internet of Things (IoT) devices, various computation-intensive mobile applications, such as speech recognition, language processing, online game and reality augmentation, are emerging. Executing computation-intensive applications poses great challenges on the MDs due to the limited battery energy and low computing power. A potential solution to address the challenges is mobile cloud computation (MCC) [1]. In MCC, the devices offload the computation tasks to remote cloud servers for execution over wireless link. Nevertheless, the execution latencies may be very large due to the long distance and the huge additional transmission load between MDs and cloud servers [2]. To handle this problem, mobile edge computing (MEC) is proposed, which reduces network latency by placing small edge servers near end users [3]. Hence MEC servers can execute and deliver the computation services rapidly to reduce the delay and save the energy consumption, which are the pivotal challenges for future radio network.

It is the common cognition in academia and industry that the efficiency of MEC is largely determined by the offloading decision [4]. Moreover, reasonable resource allocation is also important for the improvement of the performance of MEC [5]. Therefore, it is necessary to jointly optimize the policies, which comprise the offloading decision and resource allocation, to acquire the optimal solution for the delay sensitive tasks. Most of the joint optimization problems are typically NP-hard problems [6]. Owing to its non-convex property, the conventional methods used in the literatures are either exhaustive search or iterative optimization of approximation problem. Thus, the existing complexity and convergence issues of these methods may hinder their practice application. What is worse, the computation complexity of these problems would exponentially grow against the number of MDs, and the growing complexity even result in the infeasibility. In recent years, DL has achieved great success in nature language processing, speech recognition and some other fields. Some researches show that it can also be used to process hard communication issues, for instance channel precoding [7,8,9], power control [10] and channel estimation [11]. Thus, we try to tackle the MEC optimization problem by a deep-learning-based method with lower complexity. Meanwhile, computational offloading problems are generally divided into two categories: partial offloading which only offloads a subset of the task components to edge servers, and binary offloading which hands over all the task to edge servers. However, partial offloading requires to calculate the computational cost for each task component, thus puts additional work load on computation resources and energy reserves [12]. Compared with partial offloading, binary offloading is more suitable to tackle atomic tasks that are not partitionable and easier to implement in practice. Hence, it makes sense to tackle the binary offloading and resource allocation issue in MEC network with low complexity by deep-learning-based approach.

In this work, we consider a multiuser mobile-edge computation offloading (MECO) network based on Time Division Multiple Access (TDMA). To

minimize the energy comsumption of MDs, we model the MECO issue as a joint optimization problem which jointly optimizes offloading decision and resource allocation. Then, a novel unsupervised deep-learning-based BCOS is proposed to find the sub-optimal solution. Compared with the supervised deep-learning method, the unsupervised deep-learning method don't need the training dataset. Particularly, it is difficult to obtain the training dataset in this joint optimization problem due to its nonconvex nature. Moreover, this optimization problem is a mixed integer programming problem, and it causes gradient vanishing issue in DL network. To tackle this issue, we design a DNN with an auxiliary teacher network to acquire the lossless gradient information by using the auxiliary network as the teacher network in joint training phase. Then the main contributions are summarized as follows:

1. By taking latency constraint, transmit power constraint and energy consumption into account, we model the multiuser binary MECO process under TDMA system as a mixed integer programming (MIP). The optimization policies including offloading decision, transmit time slot and transmit power are jointly optimized to minimize the sum energy consumption of MDs.
2. To tackle the binary offloading issue, we propose the BCOS in which the original optimization problem is transformed as an unsupervised deep-learning problem to reduce the computation complexity. Comparing with the supervised DL, the unsupervised DL does not require the training dataset, which the is generally difficult to be obtained by solving the optimization problem with conventional mathematical method.
3. To address the gradient vanishing problem caused by binary offloading decision, we design a DNN with an auxiliary network. With the aid of the auxiliary teacher network, the student network can acquire the lossless gradient information directly. Therefore, the binary offloading problem can be solved effectively by the designed DNN. Moreover, we can obtain the sub-optimal solution by the trained DNN with low complexity.

The remainder of this paper is organized as follows. In section II, the related work of deep learning and MECO are introduced. The system model and problem formulation are proposed in section III. In section IV, binary computation offloading algorithm is presented, which includes deep learning framework, proposed deep-learning-based problem formulation and the joint training mechanism with auxiliary network. The fifth section gives the simulation results and discussions. Finally, we conclude the research work in the last section.

2 Related Works

A large amount of study works are done to improve the performance of MDs computation offloading by utilizing the advantages of edge servers [13,14]. To adapt the real-time computation offloading, several low-complexity algorithms

have been proposed to deal with the binary computation offloading problem. However, most of these works tackle the computation offloading problems with mathematical analysis fashions [2,15,16,17]. Liu et al. have presented an novel one-dimensional search algorithm to process the power-constrained delay minimization problem, which obtained the optimal offloading decision according to the queuing state of the application buffer, the available power at local processing unit and remoting transmission unit, as well as the CSI between MDs and the MEC servers [18]. However, to make offloading decision, the MDs require feedback from MEC servers in this algorithm, which increases signaling overhead additionally. In [2], a distributed computation offloading algorithm based on game theoretic method is designed to obtain efficient computation offloading decision, which requires multiple communication iterations between MDs and MEC servers. Similarly, references [19] and [20] update the binary offloading decision iteratively to solve the joint task offloading and resource allocation issue. Reference [21] propose a constrained stochastic succession convex approximation (C-SSCA) algorithm for minimizing the sum energy consumption of MDs by jointly optimizing the transmit power, offloading decision and the assignment of computation resource with low-complexity. However, all those algorithms are not applicable for the real-time computing offloading of the MEC network due to the limitation of the trade-off between computational complexity and optimality.

Machine learning has replaced traditional method in many fields, such as computer vision, natural language processing, and face recognition. The performance of DL has transcended that of the traditional machine learning methods, and DL has been widely exploited in communication fields and achieved excellent results [22,23,24,25]. The optimization framework proposed in [26] exploits deep reinforcement learning method to tackle the resource allocation in wireless MEC. In [27], a smart energy-efficient partial computation offloading scheme based on DL is proposed to minimize the cost function by selecting an offloading set which depends on the mobiles' remaining energy, energy consumption of application components, channel conditions, data size for transmission, computational load, and latency in communication. Furthermore, Gong Y et al. exploit DL method to solve the MEC offloading problem, which minimize the cost function by optimize the state of mobile environment [28]. Nevertheless, most of the aforementioned deep-learning-based methods are supervised DL and is difficult to acquire the appropriate training dataset. Hence, this motivates us to design an unsupervised deep-learning-based method without training dataset for intelligent offloading decision-making processes in multiuser MEC network.

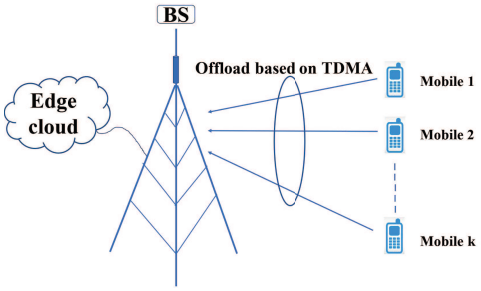

Fig. 1 Multiuser MECO system.

3 System Model And Problem Formulation

3.1 Network Model

We consider a multiuser MECO system that contain K single-antenna MDs with computation-intensive tasks and a single-antenna base station (BS) equipped with edge cloud server as shown in Fig. 1. Time is separated into slot of T seconds for K mobiles and each slot contains two phases:1) local computing or offloading and 2) cloud computing and fetching the computational results from the edge cloud server to MDs. During the time slot, the computation-intensive tasks of K MDs can be locally executed by the CPU of the terminal MDs or remotely executed via offloading to the edge servers based on TDMA. Meanwhile, we assume that the tasks are atomic and can't be split further as the strong dependence on each other. In other words, the tasks are either executed locally or completely offloaded to the edge servers. So as to enable the BS to select the offloading MDs and allocate time slots and transmit power to the offloading MDs, we also suppose that the BS masters the channel state information (CSI), the energy consumption of computing per bit and the size of task data for all users. Moreover, channels are supposed to keep unchanged within a slot.

3.2 Local Execution Model

The model of local execution is depicted as follows. We first model the computation power consumption by $p = \varepsilon f^3$,where ε is determined by the structure of chip, and f denotes the computational speed of the CPU measured by the cycle numbers per second [29]. B_k and C_k denote the size of task data and the number of cycles CPU to process 1-bit data for MD k respectively. And f_k denotes computational speed of MD k. Particularly, we assume that C_k and f_k of each mobile are fixed, which may vary over different MDs. Due to the assumption that all the tasks are atomic, we denote $a_k \in \{0, 1\}$ as the offloading strategy, $a_k = 1$ if MD k offloads its total task to edge server to compute, otherwise $a_k = 0$. Then the required number of CPU cycles for MD

k to process a B_k Kilobyte task is $(1-a_k)\,C_kB_k$ and the time consumption of local computation for MD k can be given as follow:

$$t_{k,l} = \frac{(1-a_k)C_kB_k}{f_k}. \tag{1}$$

Then the energy consumption of local computation for MD k denoted as $E_{loc,k}$, is given by

$$E_{loc,k} = p_k t_{k,l} = \varepsilon(1-a_k)B_kC_k{f_k}^2. \tag{2}$$

3.3 Offloading Model

The energy consumption of computation offloading is modeled in this subsection. Computation offloading comprises three phases for 1) uplink task transmission (i.e., wireless access to edge servers via TDMA), 2) edge execution and 3) result fetching. Assume the cloud server has infinite capacity and the tasks can run in parallel. Consequently, the latency of cloud execution is very small, and result fetching is much faster than uplink task transmission due to the relatively smaller size of the computation result. For this reason, the latency and energy consumption of edge execution and result fetching are assumed to be negligible compared with the uplink task transmission. Thus, the resource allocation of the above two phases are not considered. Then the uplink transmission rate r_k of MD k can be expressed as

$$r_k = W\log(1+\frac{p_kh_k^2}{N_0}), \tag{3}$$

where p_k and h_k refer to the transmission power and channel gain for MD k, N_0 is the variance of complex white Gaussian channel noise, W is the bandwidth. The time required for offloading task on MD k can be denoted as

$$t_k^{'} \triangleq \frac{a_kB_k}{r_k} = \frac{a_kB_k}{W\log\left(1+\frac{p_kh_k}{N_0}\right)}. \tag{4}$$

The fraction of time slot allocated to mobile k denotes as t_k, thus $t_k^{'}$ should not be larger than t_k to ensure the transmission of the completed task data. Based on the depiction above, the energy consumption of offloading $E_{off,k}$ for MD k can be expressed as

$$E_{off,k} = p_kt_k^{'}. \tag{5}$$

3.4 Problem Formulation

Our objective is to minimize the weighted sum mobile energy consumption for K MDs by adjusting the binary offloading decision $\boldsymbol{a}$, transmit time slot $\boldsymbol{t}$ and

transmit power $\boldsymbol{p}$. According to the local execution model and the offloading model, the corresponding optimization problem can be formulated as follows:

$$\begin{aligned} P1: & \min_{a_k,p_k,t_k} \sum_{k=1}^{K} \beta_k \left[p_k t_k + \varepsilon \left(1-a_k\right) B_k C_k f_k^2\right] \\ s.t. \quad & \mathrm{C1}: \sum_{k=1}^{K} t_k \leq T, \\ & \mathrm{C2}: \frac{(1-a_k)C_k B_k}{f_k} \leq T, \forall k, \\ & \mathrm{C3}: 0 \leq \mathrm{p}_k \leq p_{\max}, \forall k, \\ & \mathrm{C4}: a_k \in \{0,1\}, \forall k, \\ & \mathrm{C5}: t_k^{'} \leq t_k, \forall k, \end{aligned} \tag{6}$$

where β_k denotes the positive weight factors accounting for the fairness of MD k. Here, C1 is the time allocation constraint for K MDs. C2 denotes the latency constraint of local computation. C3 specifies the maximum transmit power constraints for per MD. C4 indicates that a task can be executed locally or offloaded to edge server for remote processing. Last, C5 ensures that a task can be offloaded completely for MD k within specified time. It can be seen that problem $P1$ is a mixed integer nonconvex problem which is NP-hard. It is challenging to solve the NP-hard problem directly by conventional mathematical analysis method [30].

4 Proposed Binary Compution Offloading Algorithm

In order to solve the NP-hard problem $P1$, the unsupervised deep-learning-based DNN model is used to implement the mapping from channel gain to offloading decision and resource allocation. In this section, we describe the binary computation offloading scheme containing details of the basic operation of the fully connected DNN, the proposed deep-learning-based problem formulation and the joint training mechanism with auxiliary network.

4.1 Deep learning framework

First, let us briefly introduce the operation of the fully connected network (FCN) in Fig. 2. The FCN used in this paper is composed of one input layer, two hidden layers and one output layer. The channel gain vector $\boldsymbol{h}$ of K MDs and optimization policies $\mathbf{a}, \mathbf{p}, \mathbf{t}$ are as the input layer and output layer respectively. The number of nodes of $i-th$ layer is denoted as $l_i, i = 1, 2$. The output of the $i-th$ hidden layer is computed as follows:

$$\boldsymbol{x}_i = \mathrm{Re}LU\left(BN\left(\boldsymbol{W}_i \boldsymbol{x}_{i-1} + \boldsymbol{b}_i\right)\right), \tag{7}$$

where $\boldsymbol{x_i}$ and $\boldsymbol{x_{i-1}}$ are the output vectors of current and force layers, and their dimensions are $l_i \times 1$ and $l_i \times 1$ respectively; $\boldsymbol{W_i}$ is the weight matrix with dimensions of $l_i \times l_{i-1}$; $\boldsymbol{b_i}$ is the bias vector with dimensions of $l_i \times 1$; ReLU

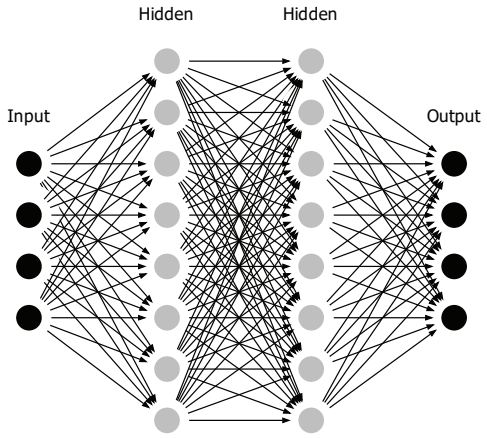

Fig. 2 A fully connected network with 2 hidden layers.

is the Rectified Linear Unit function ($\max(x, 0)$); and BN denotes the batch normalization (BN). Then, sigmoid function is chosen as the last activation function and the computation is given as follows:

$$\boldsymbol{x}_i = Sig\left(\boldsymbol{W}_i \boldsymbol{x}_{i-1} + \boldsymbol{b}_i\right), \tag{8}$$

where the sigmoid function is given as:

$$Sig\left(x\right) = \frac{1}{1 + \exp\left(-x\right)}. \tag{9}$$

Accordingly, the output of the FCN can be written with the parameterization model as

$$\boldsymbol{x_o} = \boldsymbol{x_i} = \phi(\boldsymbol{h}, \boldsymbol{\theta}), \tag{10}$$

where $\boldsymbol{h}$ is the channel gain, and $\boldsymbol{\theta}$ is the network parameters set of $\{\boldsymbol{W}, \boldsymbol{b}\}$ [31].

4.2 Proposed deep-learning-based problem formulation

Due to the standard DL issues are unconstrained issues, the approaches used to address these problems can not be adopted directly to the complex MECO problem with constraints [32]. The common approaches used to eliminate the constraints are to concatenate self-defined activation layer as output layer, or to add additional terms to the objective function as the DNN loss function for punishing the constraint violations. In this subsection, the tricks mentioned above are used to transform the origin optimization problem $P1$ into an unconstrained DL problem.

First, the DNN used to address the optimization problem $P1$ is comprised of three parallel FCNs. For this DNN, the input is the channel gain $\boldsymbol{h}$ and the

output is the solution of problem $P1$, namely, the offloading decision $\boldsymbol{a}$, the time allocation $\boldsymbol{t}$ and the transmit power allocation $\boldsymbol{p}$. According the equation (10), we employ the near universal parametrization method to parameterize the offloading decision and the resource allocation function:

$$\begin{aligned} \boldsymbol{a} &= \phi_a(\boldsymbol{h}, \boldsymbol{\theta}), \\ \boldsymbol{t} &= \phi_t(\boldsymbol{h}, \boldsymbol{\theta}), \\ \boldsymbol{p} &= \phi_p(\boldsymbol{h}, \boldsymbol{\theta}), \end{aligned} \tag{11}$$

where $\boldsymbol{h}$ is the input channel gain, and $\boldsymbol{\theta}$ is the network parameters. Then the problem $P1$ can be modified as:

$$\begin{aligned} P2: \min_{\theta} & \textstyle\sum_{k=1}^{K} \beta_k [\phi_p^k(\boldsymbol{h}, \boldsymbol{\theta}) \phi_t^k(\boldsymbol{h}, \boldsymbol{\theta}) \\ & + \varepsilon \left(1 - \phi_a^k(\boldsymbol{h}, \boldsymbol{\theta})\right) B_k C_k f_k^2] \\ s.t. \quad & \mathrm{C1}: \textstyle\sum_{k=1}^{K} \phi_t^k(\boldsymbol{h}, \boldsymbol{\theta}) \leq T, \\ & \mathrm{C2}: \frac{\left(1 - \phi_a^k(\boldsymbol{h}, \boldsymbol{\theta})\right) C_k B_k}{f_k} \leq T, \forall k, \\ & \mathrm{C3}: 0 \leq \phi_p^k(\boldsymbol{h}, \boldsymbol{\theta}) \leq p_{\max}, \forall k, \\ & \mathrm{C4}: \phi_a^k(\boldsymbol{h}, \boldsymbol{\theta}) \in \{0, 1\}, \forall k, \\ & \mathrm{C5}: t_k^{'} \leq \phi_t^k(\boldsymbol{h}, \boldsymbol{\theta}), \forall k, \end{aligned} \tag{12}$$

where $a_k = \phi_a^k(\boldsymbol{h}, \boldsymbol{\theta})$, $p_k = \phi_p^k(\boldsymbol{h}, \boldsymbol{\theta})$ and $t_k = \phi_t^k(\boldsymbol{h}, \boldsymbol{\theta})$ denotes the sub-optimal policies for MD k. Although the problem $P2$ is non-convex, the loss of the optimality is small for the near-universal parameterizations according to the Theorem1 in [31].

To meet the constraints of the offloaded decision $\boldsymbol{a}$ and resources allocations $\boldsymbol{p}$ and $\boldsymbol{t}$, we concatenate different self-defined activation layers at the end of the FCN, and $\boldsymbol{x}_o$ in equation (10) is the input of self-defined layers. Then, the different self-defined layers are given respectively as follows:

1. To satisfy the transmit power constraint C3 in problem $P2$, the activation function of the last layer for the FCN is expressed as:

$$\boldsymbol{p} = p_{\max} \phi(\boldsymbol{h}, \boldsymbol{\theta}). \tag{13}$$

2. To satisfy the time allocation constraint C1 in problem $P2$, the normalization activation function of the last layer for the FCN is given as:

$$\boldsymbol{t} = T \min(1, ||\phi(\boldsymbol{h}, \boldsymbol{\theta})||) \frac{\phi(\boldsymbol{h}, \boldsymbol{\theta})}{||\phi(\boldsymbol{h}, \boldsymbol{\theta})||}. \tag{14}$$

3. The binary offloading decision need to satisfy the constraints C2 and C4. Namely, if the whole task can not be completed in the specified time locally, it must be offloaded to edge servers for execution. In light of constraint C4, we can obtain a ratio $a_k \geq m_k^+$ with $m_k = 1 - \frac{f_k T}{C_k B_k}$ and $x^+ = \max\{x, 0\}$ for MD k, which denotes the ratio of task beyond local computing power.

Once the ratio is greater than 0, then $a_k = 1$. Accordingly, the activation function of the last layer for FCN can be given as:

$$\boldsymbol{a} = sign\left[\frac{sign\left(2\phi(\boldsymbol{h},\boldsymbol{\theta})-1\right)+1}{2} + sign\left(\boldsymbol{m_k^+}\right)\right], \tag{15}$$

where the vector $\boldsymbol{m}_k^+$ denotes the task ratio vector, and the sign function is given as follows:

$$sign\left(x\right) = \begin{cases} -1, x < 0 \\ 0, x = 0 \\ 1, x > 0. \end{cases} \tag{16}$$

As a result, the constraints C1, C2, C3 and C4 are addressed by the activation layers at the end of the FCN. Further, the optimization problem $P2$ can be transformed as follows:

$$\begin{aligned} P3: \min_{\theta} & \textstyle\sum_{k=1}^{K} \beta_k[\phi_p^k(\boldsymbol{h},\boldsymbol{\theta})\phi_t^k(\boldsymbol{h},\boldsymbol{\theta}) \\ & +\varepsilon\left(1-\phi_a^k(\boldsymbol{h},\boldsymbol{\theta})\right)B_k C_k f_k^2] \\ s.t. \quad & \mathrm{C5}: t_k^{'} - \phi_t^k(\boldsymbol{h},\boldsymbol{\theta}) \leq 0, \forall k. \end{aligned} \tag{17}$$

However, the optimization problem $P3$ is still constrained. To remove the constraint C5, the penalty term is introduced in the loss function of DNN. Since we are merely focus on eliminating the states that dissatisfy the constraints, we can neglect the value of function $t_k^{'} - \phi_t^k(\boldsymbol{h},\boldsymbol{\theta})$ when constraint C5 is satisfied. Consequently, the Hinge function is defined as follows:

$$H\left(c\right) = \begin{cases} c, c \geq 0 \\ 0, c < 0. \end{cases} \tag{18}$$

The constraint C5 can be replaced equally by $\mathrm{H}(t_k^{'} - \phi_t^k(\boldsymbol{h},\boldsymbol{\theta})) = 0, \forall k$ without changing the origin formulation. Specifically, $\mathrm{H}(t_k^{'} - \phi_t^k(\boldsymbol{h},\boldsymbol{\theta}))$ can be thought as the loss when constraint C5 is dissatisfied. If the constraint is satisfied, the loss is zero. Hence, the optimization problem $P2$ can be transformed to minimize a new loss function of DNN which penalizes the constraint violation. That is, the overall learning problem can be expressed as:

$$P3: \quad \min_{\boldsymbol{\theta},\lambda_B} \quad loss_{binary}\left(\boldsymbol{\theta},\lambda_B\right), \tag{19}$$

where the loss function can be expressed as:

$$\begin{aligned} loss_{binary} = E\Big\{ & \textstyle\sum_{k=1}^{K} \beta_k[\phi_p^k(\boldsymbol{h},\boldsymbol{\theta})\phi_t^k(\boldsymbol{h},\boldsymbol{\theta}) \\ & +\varepsilon(1-\phi_a^k(\boldsymbol{h},\boldsymbol{\theta}))] \\ & +\lambda_B \textstyle\sum_{k=1}^{K} H(t_k^{'} - \phi_t^k(\boldsymbol{h},\boldsymbol{\theta}))\Big\}. \end{aligned} \tag{20}$$

In the loss function (20), to make the DNN output satisfy the offloading time constraints, penalty term is introduced. If $t_k^{'} > \phi_t^k(\boldsymbol{h},\boldsymbol{\theta})$, then $H\left(t_k^{'} - \phi_t^k(\boldsymbol{h},\boldsymbol{\theta})\right) >$

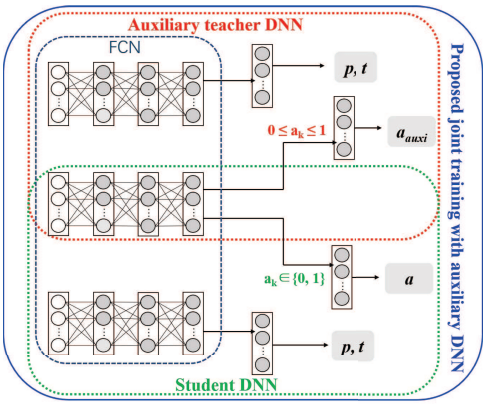

Fig. 3 Schematic of the proposed joint training with auxiliary DNN. The FCN of offloading decision is shared by the auxiliary teacher network and the student network. In training phase, the parameters of the auxiliary teacher network and the student network are updated alternatively. Then, the shared FCN can obtain the lossless gradient information during backpropagation.

0, namely, the offloading time constraints are not satisfied. To minimize the loss function, the penalty term compels the DNN updating along the direction of constraint satisfaction. In the opposite, if $t_k^{'} \leq \phi_t^k(\boldsymbol{h}, \boldsymbol{\theta})$, $H\left(t_k^{'} - \phi_t^k(\boldsymbol{h}, \boldsymbol{\theta})\right) = 0$ and the penalty term has no effect on the loss function. In this instance, the training process concentrates on the satisfaction of other MDs and the minimization of the energy consumption of K MDs. λ_B is the scaling factor, which is used to balance the gap between different terms of the loss function. Therefore, λ_B needs to be adjusted carefully as a hyperparameters: if too small, the DNN may minimize the energy consumption mainly and output an infeasible solution violated the constraint; if too large, the DNN may concentrates on the satisfaction of the constraints while neglecting the minimization of the sum energy consumption. It is generally difficult to select a suitable hyperparameter in DL and many optimization problems. In this work, we use sub-gradients to handle the non-differentiability caused by the Hinge function, and the parameter update equation of λ_B can be written as:

$$\lambda_B^{(i+1)} \leftarrow \lambda_B^{(i)} + \alpha \nabla_{\lambda_B} loss_{binary}(\boldsymbol{\theta}, \lambda_B), \tag{21}$$

where

$$\nabla_{\lambda_B} loss_{binary}(\boldsymbol{\theta}, \lambda_B) = \sum\nolimits_{k=1}^{K} H(t_k^{'} - \phi_t^k(\boldsymbol{h}, \boldsymbol{\theta})). \tag{22}$$

4.3 Joint training mechanism with auxiliary network

Although the problem $P3$ is an unconstrained optimization problem, the binary offloading decision considered in the MECO network will result in gradient vanishing issue in the training process. In other words, the back-propagation

method can not effectively update the gradients of the neural layers before the binary layer when the non-differentiable operators are adopted. To address this issue, a DNN with an auxiliary network is designed in Fig. 3. In this subsection, we introduce the training process of the proposed DNN in detail.

To map the channel gain to a binary offloading decision of 0 or 1, the common binarization operators (e.g., sign()) may be applied in the activation function of DNN. While the fact that the derivative of the output of the binarization operator neuron is zero everywhere except the origin where the function is non-differentiable, may result in the vanishing gradient problem during backpropagation. The most common practice to overcome this problem is to approximate the activation function of binarization layer with a smooth differentiable function during backpropagation. Such an approximation of a binarization layer during backpropagation is taken as straight-though estimation (STE), which is first proposed by G. E. Hinton [33,34]. However, the approximation may cause noisy signal when updating the parameters of DNN due to the incorrect updating direction [35,36].

To mitigate this, we design the DNN containing an auxiliary teacher network and a student network in Fig. 3. Moreover, we suppose that the tasks can be tailored and the offloading decision constraint C4 are relaxed as $0 \leq a_{auxi}^k \leq 1, \forall k$ for the auxiliary network. Thus, a_{auxi}^k denotes the offload ratio of task on mobile device k. To satisfy this constraint, the last layer activation function of auxiliary network of offloading decision is defined as:

$$\boldsymbol{a}_{auxi} = \phi(\boldsymbol{h}, \boldsymbol{\theta})(1 - \boldsymbol{m}_k^+) + \boldsymbol{m}_k^+. \tag{23}$$

Therefore, the student network and the auxiliary teacher network have the same structure except the different activation function layer for the offloading decision. The identical structure is effective to transmit information from the auxiliary network to the student network and reduce the information loss caused by the structure mismatch between the auxiliary network and the student network[9]. Additionally, the network before the last activation layer for the offloading decision is shared by the auxiliary network (in the red line box) and the student network (in the green line box) as seen in Fig. 3. Thus, the student network can directly acquire the lossless gradient from the auxiliary network.

Similar to the learning problem $P3$, the loss function of the auxiliary network can be given as:

$$P4: \quad \min_{\boldsymbol{\theta}_{auxi}, \lambda_A} loss_{auxi}\left(\boldsymbol{\theta}_{auxi}, \lambda_A\right), \tag{24}$$

where the loss function can be expressed as:

$$\begin{aligned} loss_{auxi} = E\Big\{ & \textstyle\sum_{k=1}^{K} \beta_k [\phi_p^k(\boldsymbol{h}, \boldsymbol{\theta}_{auxi}) \phi_t^k(\boldsymbol{h}, \boldsymbol{\theta}_{auxi}) \\ & + \varepsilon(1 - \phi_a^k(\boldsymbol{h}, \boldsymbol{\theta}_{auxi}))] \\ & + \lambda_A \textstyle\sum_{k=1}^{K} H(t_k^{'} - \phi_t^k(\boldsymbol{h}, \boldsymbol{\theta}_{auxi}))\Big\}. \end{aligned} \tag{25}$$

To utilize the auxiliary network in training phase, we exploit a joint training approach to alternatively train the auxiliary network and the student network [33, 34]. The detailed procedure is presented in Algorithm 1. The auxiliary network and the student network are sequentially updated in each iteration process. Therefore, the teacher network can guide the student network effectively to obtain lossless information for generalization, and the corresponding joint training prohibits the student network from being trapped at a poor local minimum [33-35].

Algorithm 1 Joint Training the Proposed DNN

1: **Initialize** $\boldsymbol{\theta}, \boldsymbol{\theta}_{auxi}, \lambda_B, \lambda_A, \alpha$.
2: Generate 20000 channel data samples with the minibatch-size 1000.
3: **for** $i = 1$:num_iterations **do**
4: Update teacher network parameters $\boldsymbol{\theta}_{auxi}$ by minimizing $loss_{auxi}(\boldsymbol{\theta}_{auxi}, \lambda_A)$ with ADAM.
5: Update student network parameters $\boldsymbol{\theta}$ by minimizing $loss_{binary}(\boldsymbol{\theta}, \lambda_B)$ with ADAM.
6: Update penalty terms hyperparameters λ_A and λ_B by $\lambda_A^{(i+1)} \leftarrow \lambda_A^{(i)} + \alpha \nabla_{\lambda_A} loss_{auxi}(\boldsymbol{\theta}_{auxi}, \lambda_A)$ and $\lambda_B^{(i+1)} \leftarrow \lambda_B^{(i)} + \alpha \nabla_{\lambda_B} loss_{binary}(\boldsymbol{\theta}, \lambda_B)$.
7: **end for**

5 Numerical Simulation

In this section, numerical simulations are carried out to evaluate the efficiency of BCOS. We consider a multiuser MEC system comprising of a single-antenna BS equipped with edge servers and K MDs. We model the channel as independent Rayleigh fading and set the large-scale fading average power loss as 10^{-6}. Other simulation parameters are shown in Table 1 unless specified otherwise.

To evaluate the performance of our proposed BCOS, we take three offloading schemes into account and explain them particularly as follows:

1. Minimum computation offloading scheme (MCOS): In this approach, we give priority to local execution. If the local computation capacity is insufficient, all of the remainder will be offloaded with maximum transmitting power to edge servers and the weighted sum energy consumption of K MDs is computed.
2. Partial computation offloading scheme (PCOS): Assume that the tasks can be partitioned arbitrarily, we use the single teacher network to optimize the offloading ratio, transmitting power, and transmitting time of tasks for minimizing the energy consumption of K MDs by unsupervised DL method.

Table 1 Simulation Parameters

Notation	Parameter	Value
W	Total bandwidth	10MHz
K	Number of mobiles	4-20
P_{max}	Maximum transmit power	23dBm
N_0	Variance of complex White Gaussian noise	10-9W
B_k	Size of task input data	100-300KB
C_k	CPU cycles per bit	200-400
T	Time slot	0.5S
f_k	Local computation capacity	1-2GHz
β_k	Equal fairness weight factors	1
ε	Chip parameter	$1e^{-28}$

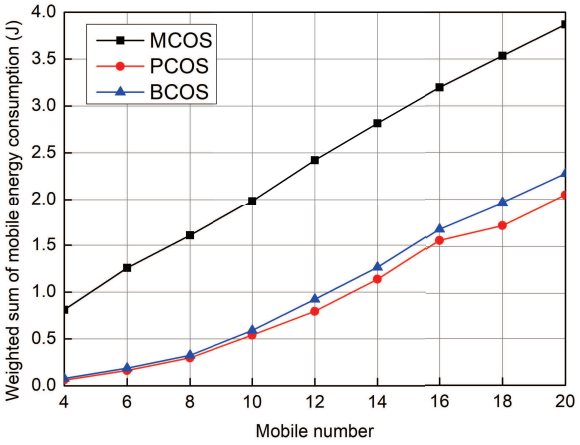

Fig. 4 The comparison of weighted sum mobile energy consumption between MCOS, PCOS and BCOS with the increasement of mobile number from 4 to 20, where $T = 0.5s$.

3. Binary computation offloading scheme (BCOS): The task which is not partitioned must be tackled all at local or at edge servers. We optimize the binary offloading decision, transmitting power and transmitting time for minimizing the energy consumption of K MDs by unsupervised DL method.

The weighted sum energy consumption of the three different schemes for K MDs are compared in Fig. 4 with the increase of MD number from 4 to 20, where $T = 0.5s$. We can find that for all three schemes, the weighted sum energy consumption increases with the MD number. The energy consumption of MCOS is always more than the other two schemes, indicating that the computation offloading is effective to save the energy consumption of MDs.

In addition, the energy consumption of PCOS and BCOS are relatively close. When the number of MDs is less than 8, the energy consumption of PCOS and BCOS are approximately the same. While MD number is more than 8, the gap of energy consumption between PCOS and BCOS begin to increase. This is because that when the number of MD is small, the time

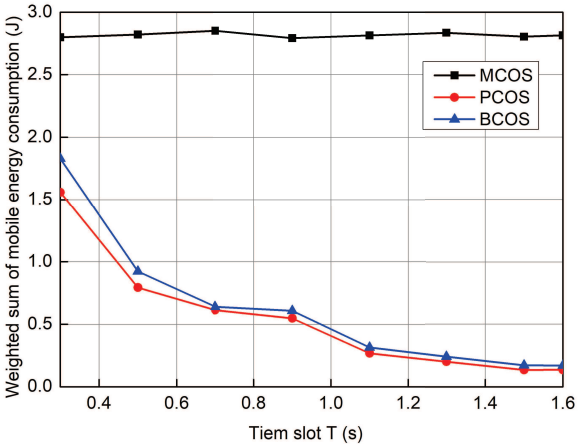

Fig. 5 The comparison of weighted sum mobile energy consumption between MCOS, PCOS and BCOS with the increasement of time slot T from 0.3s to 1.6s, where the mobile number is 14.

deadline is relatively loose, and the devices can be allocated enough time to offload all the tasks. With the increase of mobile device number, the time deadline is tightened and the tasks may need to be tailored. In view of the task integrity, BCOS can't offload as many tasks as PCOS does.

We further investigate the effect of communication resource on the weighted sum energy consumption. The comparison of weighted sum mobile energy consumption between MCOS, PCOS and BCOS with the increasement of time slot T from 0.3s to 1.6s is shown in Fig. 5, where $K = 14$. Compared with another two schemes, the energy savings of the proposed BCOS are evaluated. With the growth of time slot T, the energy consumption of PCOS and BCOS first decrease and then stabilize when $T \geq 1.5s$, while the energy consumption of MCOS remains almost unchanged. This is because that the increased time slots allow more MDs to be selected to offload their tasks. when time slot T is greater than 1.5s, the energy consumption of PCOS and BCOS are not abating. It indicates that the allocated time has reached saturation and meets the requirement of offloading all the tasks. Therefore, the further increase of time slot T has no effect on energy consumption. For MCOS, it gives priority to local computation and the assigned time always meets the requirement of offloading the remainder task.

Furthermore, we can see that the energy consumption of PCOS is slightly lower than that of BCOS at the beginning and then the gap become smaller with the increasement of T. Due to that the partial offloading decision is exploited by PCOS, the tasks on MD can be tailored flexibly and offloaded as much as possible. However, PCOS violates the integrity of the task. Hence, the relaxed latency constraint can help to reduce the difference between binary offloading and partial offloading, since the tasks can be offloaded as a whole without compromising the integrity.

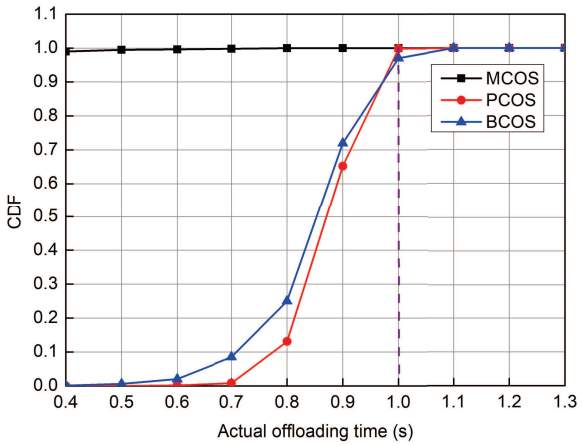

Fig. 6 CDF of the sum of realistic task offloading time for K MDs where $T = 1s$ and $K = 14$.

Finally, we depict the cumulative distribution function (CDF) of the sum of the realistic task offloading time for K MDs in Fig. 6, where $T = 1s$ and $K = 14$. It is seen that for MCOS, all MD have a high probability of the realistic task execution delay at 0.5s, which is much lower than the prescribed delay constraint 1s. For the reason that MCOS gives priority to local execution, only a small number of tasks are offloaded to edge servers, and the required time is relatively less. However, both PCOS and BCOS have a few samples whose realistic offloading time is beyond the prescribed time constraint. It is because that the number of the samples violated the delay constraint is affected by hyperparameters λ_B and λ_A of penalty terms in loss function (20) and (25). The hyperparameters are used to balance the objective function and realistic offloading time constraint in loss function. If too large, the DNN would concentrate on meeting the realistic offloading time constraint and the number of samples violated the delay constraint would be further reduced or even become 0 while sacrificing the sum energy consumption; If too small, the DNN would mainly minimize the weighted sum energy consumption and output an infeasible solution with the increased number of samples violated the delay constraint. Hence, there are a few samples violated the delay constraint obtain the sub-optimal offloading decisions and the lowest energy consumption. On the whole, both PCOS and BCOS complete the offloading tasks with a very high probability within the prescribed delay.

6 Conclusion

In this study, we have proposed an unsupervised DL binary offloading problem for computation intensive tasks on MD in multiuser MEC network to save the MD energy consumption. The problem is formulated as an optimization issue, which jointly optimize the binary offloading decision, transmit time and

transmit power, to minimize the energy consumption of K MDs under the constraints of latency and transmit power. Due to the binary offloading decision, the optimization issue is a MIP problem, and the gradient vanishing problem occurs in backpropagation. To tackle this, we have proposed the BCOS, in which we design a DNN with a auxiliary network. With the assistance of the auxiliary network, the student network acquire the gradient information without loss. Finally, the sub-optimal solution of the optimization issue is obtained by the unsupervised DL based BCOS. The simulation results indicate that the mobile edge computation offloading approach is effective to reduce the energy consumption of MDs. Moreover, both BCOS and PCOS can solve the optimization problem effectively with binary offloading decision and partial offloading decision respectively by trained DNN with low complexity. Especially, the binary offloading decision is suitable for the sample tasks which can't be tailored. In the future, we will expand the proposed MEC network model to consider the computation resource allocation of edge servers, multi-antenna access point and interference channel, etc.

Declaration of Competing Interest

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

Acknowledgements

This work was financially supported by the Fundamental Research Funds for the Central Universities (No. CCNU20TS008).

References

1. Fu X, Secci S, Huang D, Jana R. (2015). Mobile Cloud Computing. *IEEE Communications Magazine*, 53(3):61-2.
2. Chen X, Jiao L, Li W, Fu X. (2016). Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing. *IEEE/ACM Transactions on Networking*, 24(5):2795-808.
3. Xu J, Wang S, Bhargava BK, Yang F. (2019). A Blockchain-Enabled Trustless Crowd-Intelligence Ecosystem on Mobile Edge Computing. *IEEE Transactions on Industrial Informatics*, 15(6):3538-47.
4. Muñoz O, Pascual-Iserte A, Vidal J. (2015). Optimization of Radio and Computational Resources for Energy Efficiency in Latency-Constrained Application Offloading. *IEEE Transactions on Vehicular Technology*, 64(10):4738-55.
5. Sardellitti S, Scutari G, Barbarossa S. (2015). Joint Optimization of Radio and Computational Resources for Multicell Mobile-Edge Computing. *IEEE Transactions on Signal and Information Processing over Networks*, 1(2):89-103.
6. Lyu X, Tian H, Sengul C, Zhang P. (2017). Multiuser Joint Task Offloading and Resource Optimization in Proximate Clouds. *IEEE Transactions on Vehicular Technology*, 66(4):3435-47.
7. Nachmani E, Marciano E, Lugosch L, Gross WJ, Burshtein D, Be'ery Y. (2018). Deep Learning Methods for Improved Decoding of Linear Codes. *IEEE Journal of Selected Topics in Signal Processing*, 12(1):119-31.
8. Liang F, Shen C, Wu F. (2018). An Iterative BP-CNN Architecture for Channel Decoding. *IEEE Journal of Selected Topics in Signal Processing*, 12(1):144-59.

9. Kong K, Song WJ, Min M. (2020). Deep-learning-based precoding in multiuser MIMO downlink channels with limited feedback. arXiv preprint arXiv:2008.04147.
10. Liang F, Shen C, Yu W, Wu F. (2020). Towards Optimal Power Control via Ensembling Deep Neural Networks. *IEEE Transactions on Communications*, 68(3):1760-76.
11. Wei X, Hu C, Dai L. (2021). Deep Learning for Beamspace Channel Estimation in Millimeter-Wave Massive MIMO Systems. *IEEE Transactions on Communications*, 69(1):182-93.
12. Orsini G, Bade D, Lamersdorf W. (2016). CloudAware: A Context-Adaptive Middleware for Mobile Edge and Cloud Computing Applications. *2016 IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS* W)*, p. 216-221.
13. Kai C, Zhou H, Yi Y, Huang W. (2020). Collaborative Cloud-Edge-End Task Offloading in Mobile-Edge Computing Networks with Limited Communication Capability. *IEEE Transactions on Cognitive Communications and Networking*. early access, doi: 10.1109/TCCN.2020.3018159..
14. Li M, Gao J, Zhao L, Shen X. (2020). Deep Reinforcement Learning for Collaborative Edge Computing in Vehicular Networks. *IEEE Transactions on Cognitive Communications and Networking*, 6(4):1122-35.
15. Cui Y, Zhang D, Zhang T, Chen L, Piao M, Zhu H. (2020). Novel method of mobile edge computation offloading based on evolutionary game strategy for IoT devices. *AEU - International Journal of Electronics and Communications*, 118(153134).
16. Xu X, Zhang X, Gao H, Xue Y, Qi L, Dou W. (2020). BeCome: Blockchain-Enabled Computation Offloading for IoT in Mobile Edge Computing. *IEEE Transactions on Industrial Informatics*, 16(6):4187-95.
17. Zhao J, Li Q, Gong Y, Zhang K. (2019). Computation Offloading and Resource Allocation For Cloud Assisted Mobile Edge Computing in Vehicular Networks. *IEEE Transactions on Vehicular Technology*, 68(8):7944-56.
18. Liu J, Mao Y, Zhang J, Letaief KB. (2016). Delay-optimal computation task scheduling for mobile-edge computing systems. *2016 IEEE International Symposium on Information Theory (ISIT)*, 1451-5.
19. Tran TX, Pompili D. (2019). Joint Task Offloading and Resource Allocation for Multi-Server Mobile-Edge Computing Networks. *IEEE Transactions on Vehicular Technology*, 68(1):856-68.
20. Bi S, Zhang YJ. (2018). Computation Rate Maximization for Wireless Powered Mobile-Edge Computing With Binary Computation Offloading. *IEEE Transactions on Wireless Communications*, 17(6):4177-90.
21. Wang J, Feng D, Zhang S, Liu A, Xia XG. (2020). Joint Computation Offloading and Resource Allocation for MEC-enabled IoT Systems with Imperfect CSI. *IEEE Internet of Things Journal*, 8(5):3462-75.
22. Zhang C, Patras P, Haddadi H. (2019). Deep Learning in Mobile and Wireless Networking: A Survey. *IEEE Communications Surveys & Tutorials*, 21(3):2224-87.
23. Sun H, Chen X, Shi Q, Hong M, Fu X, Sidiropoulos ND. (2017). Learning to optimize: Training deep neural networks for wireless resource management. *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, p. 1-6.
24. Gao Z, Jiao Q, Xiao K, Wang Q, Mo Z, Yang Y. (2019). Deep Reinforcement Learning Based Service Migration Strategy for Edge Computing. *2019 IEEE International Conference on Service-Oriented System Engineering (SOSE)*. p. 116-1165.
25. Samuel N, Diskin T, Wiesel A. (2017). Deep MIMO detection. p. 1-5.
26. Li J, Gao H, Lv T, Lu Y. (2018). Deep reinforcement learning based computation offloading and resource allocation for MEC. *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, p. 1-6.
27. Ali Z, Jiao L, Baker T, Abbas G, Abbas ZH, Khaf S. (2019). A Deep Learning Approach for Energy Efficient Computational Offloading in Mobile Edge Computing. *IEEE Access*, 7(149623-33).
28. Gong Y, Lv C, Cao S, Yan L, Wang H. (2020). Deep learning-based computation offloading with energy and performance optimization. *EURASIP Journal on Wireless Communications and Networking*, 2020(1):69.

29. Wang Y, Sheng M, Wang X, Wang L, Han W, Zhang Y, et al. (2015). Energy-optimal partial computation offloading using dynamic voltage scaling. *2015 IEEE International Conference on Communication Workshop (ICCW)*, p. 2695-700.
30. Pochet Y, Wolsey LA, (2006). Production Planning by Mixed Integer Programming. *Springer*.
31. Eisen M, Zhang C, Chamon LFO, Lee DD, Ribeiro A. (2019). Learning Optimal Resource Allocations in Wireless Systems. *IEEE Transactions on Signal Processing*, 67(10):2775-90.
32. Eisen M, Zhang C, Chamon LFO, Lee DD, Ribeiro A. (2019). Dual Domain Learning of Optimal Resource Allocations in Wireless Systems. *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, p. 4729-33.
33. Bengio Y. (2013). Estimating or Propagating Gradients Through Stochastic Neurons. *Universite de Montreal*, Technical Report arXiv:1305.2982.
34. Zhuang B, Shen C, Tan M, Liu L, Reid I. (2018). Towards Effective Low-Bitwidth Convolutional Neural Networks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, p. 7920-8.
35. Zhuang B, Liu J, Tan M, Liu L, Shen C. (2019). Effective Training of Convolutional Neural Networks with Low-bitwidth Weights and Activations. arXiv preprint arXiv:1908.04680.
36. Zhuang B, Liu L, Tan M, Shen C, Reid I. (2020). Training Quantized Neural Networks With a Full-Precision Auxiliary Module. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, p. 1488-97.

Figures

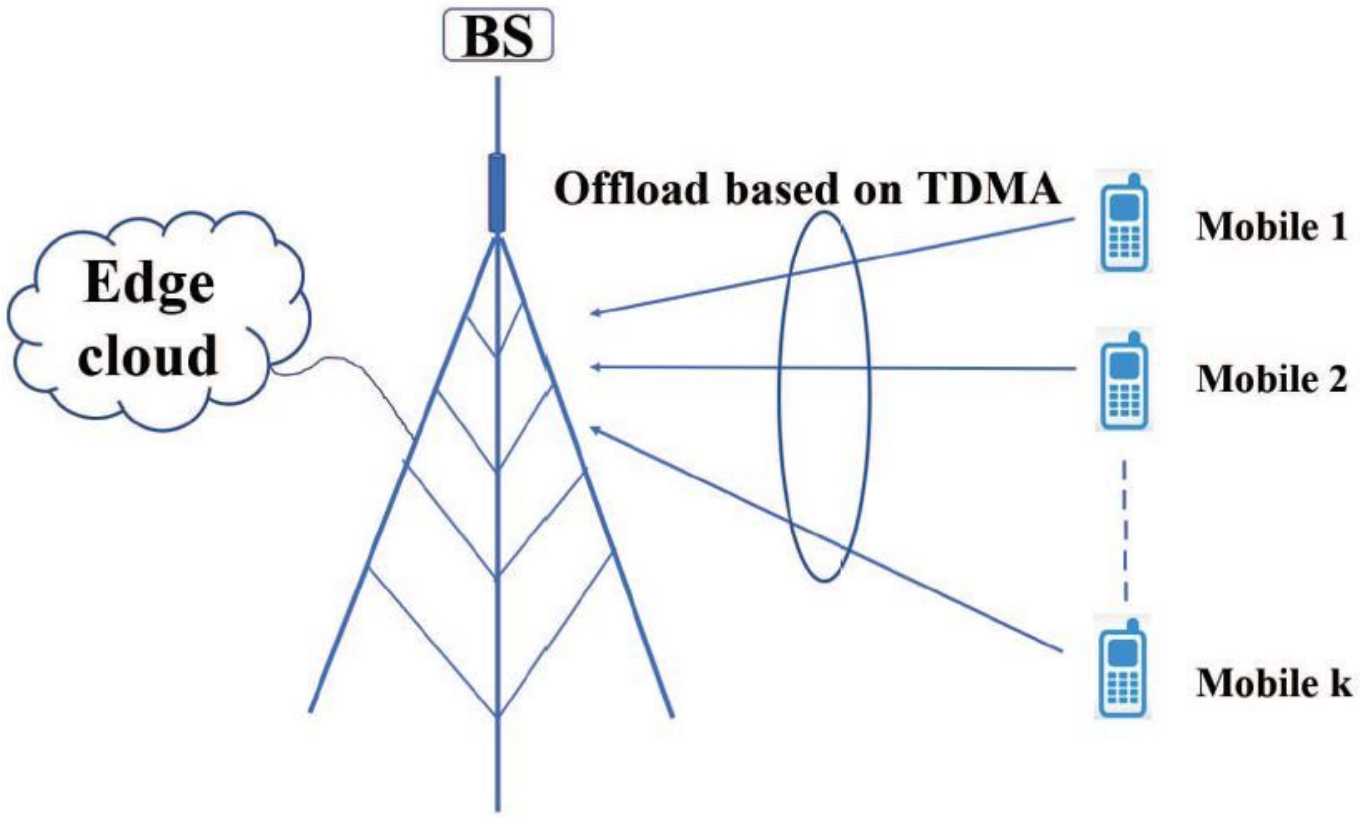

Figure 1

Multiuser MECO system.

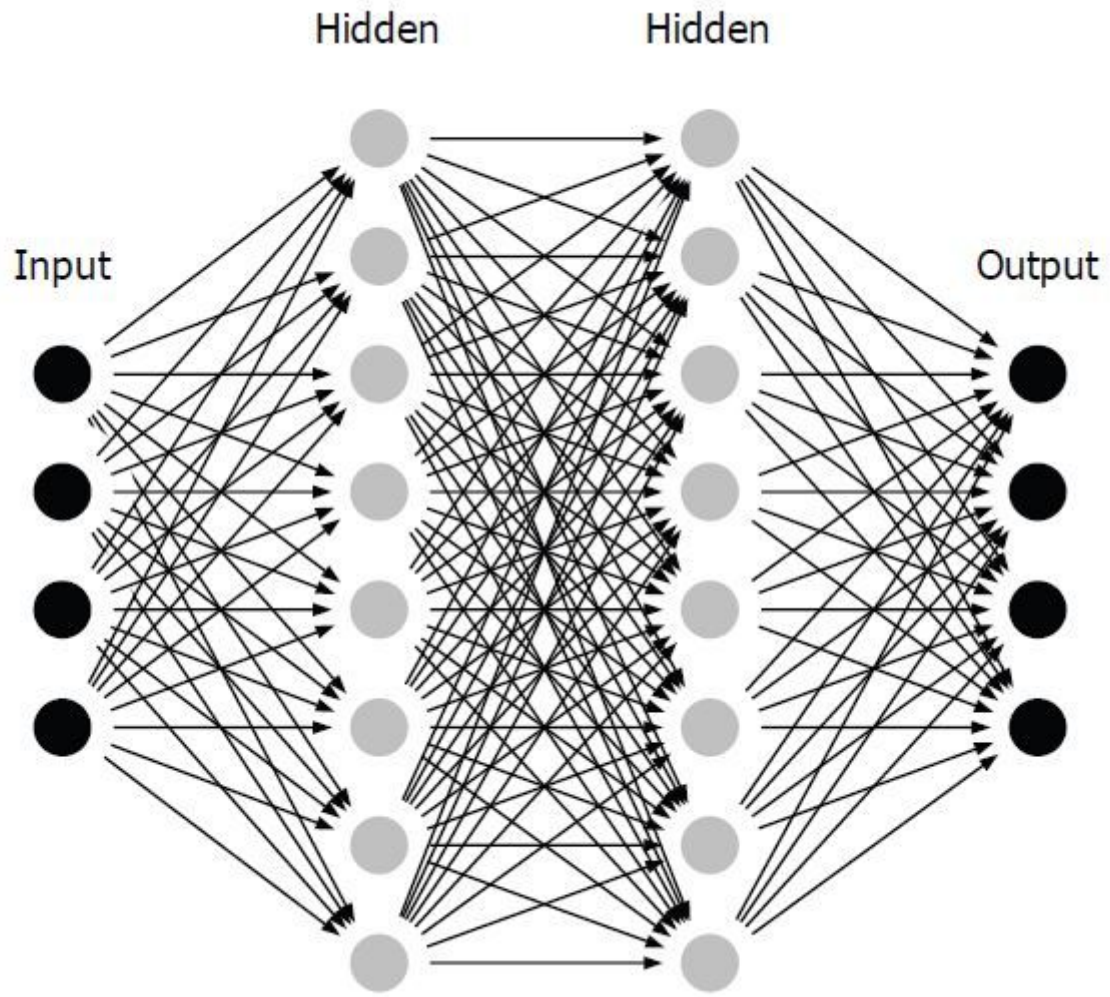

Figure 2

A fully connected network with 2 hidden layers.

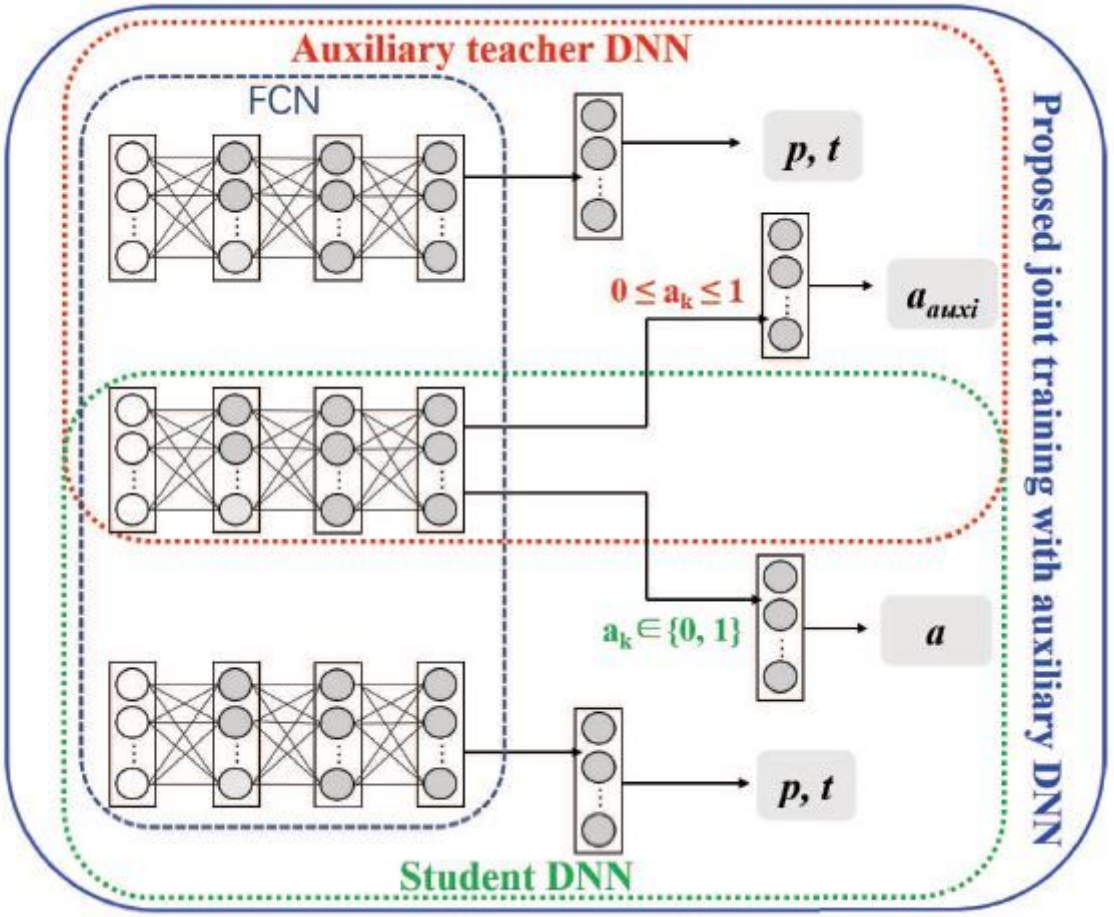

Figure 3

Schematic of the proposed joint training with auxiliary DNN. The FCN of offloading decision is shared by the auxiliary teacher network and the student network. In training phase, the parameters of the auxiliary teacher network and the student network are updated alternatively. Then, the shared FCN can obtain the lossless gradient information during backpropagation.

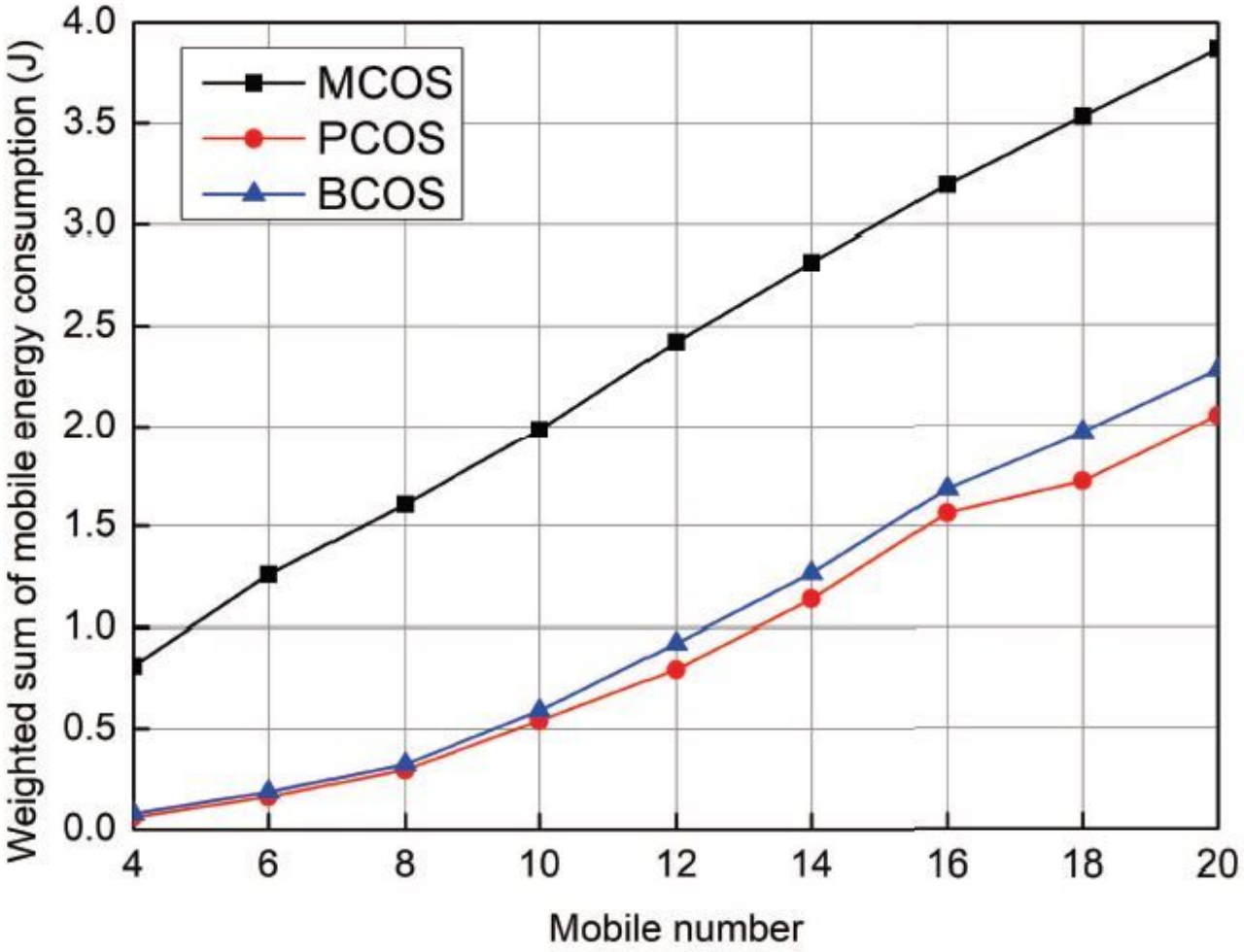

Figure 4

The comparison of weighted sum mobile energy consumption between MCOS, PCOS and BCOS with the increasement of mobile number from 4 to 20, where T = 0:5s.

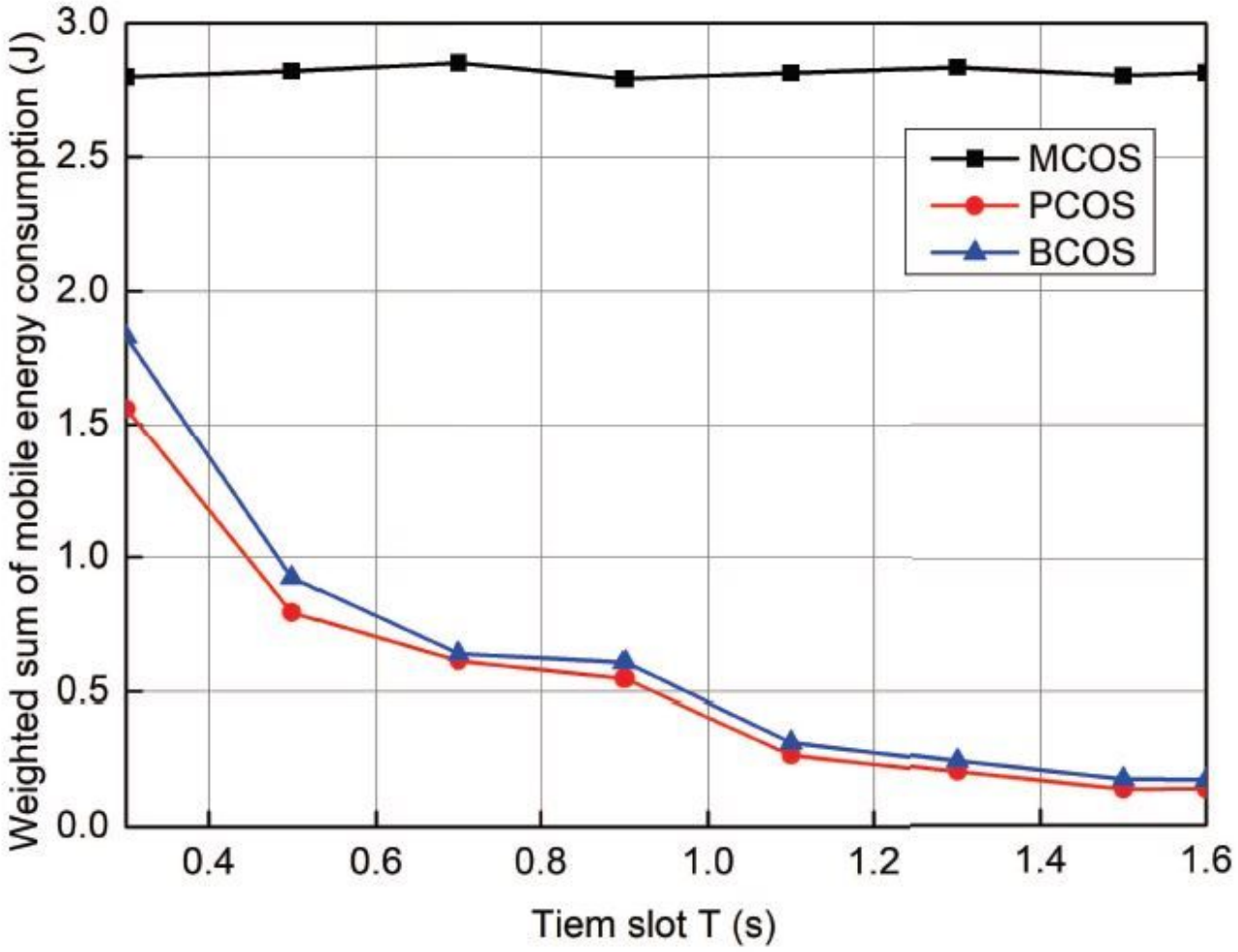

Figure 5

The comparison of weighted sum mobile energy consumption between MCOS, PCOS and BCOS with the increasement of time slot T from 0.3s to 1.6s, where the mobile number is 14.

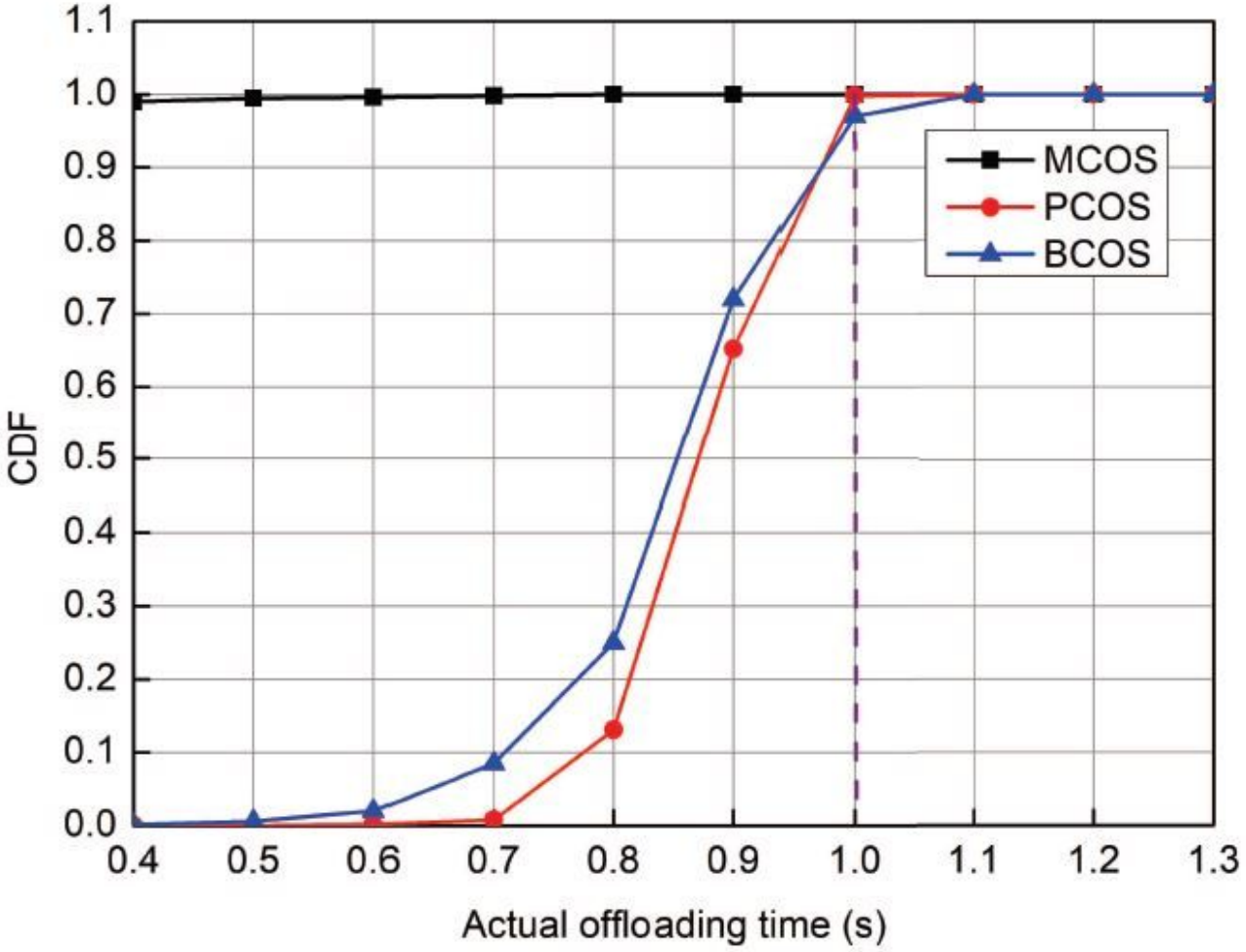

Figure 6

CDF of the sum of realistic task offloading time for K MDs where T = 1s and K = 14.