Trust-Based Social Networks with Computing, Caching and Communications: A Deep Reinforcement Learning Approach

Ying He, *Student Member, IEEE*, Chengchao Liang, *Member, IEEE*, F. Richard Yu, *Fellow, IEEE*, and Zhu Han, *Fellow, IEEE*

Abstract—Social networks have continuously been expanding and trying to be innovative. The recent advances of computing, caching, and communication (3C) can have significant impacts on mobile social networks (MSNs). MSNs can leverage these new paradigms to provide a new mechanism for users to share resources (e.g., information, computation-based services). In this paper, we exploit the intrinsic nature of social networks, i.e., the trust formed through social relationships among users, to enable users to share resources under the framework of 3C. Specifically, we consider the mobile edge computing (MEC), in-network caching and device-to-device (D2D) communications. When considering the trust-based MSNs with MEC, caching and D2D, we apply a novel deep reinforcement learning approach to automatically make a decision for optimally allocating the network resources. The decision is made purely through observing the network's states, rather than any handcrafted or explicit control rules, which makes it adaptive to variable network conditions. Google TensorFlow is used to implement the proposed deep Q-learning approach. Simulation results with different network parameters are presented to show the effectiveness of the proposed scheme.

Index Terms—Mobile social networks, mobile edge computing, caching, deep reinforcement learning

1 Introduction

Mobile social networks (MSNs) have been developing rapidly, which can provide a variety of social services and applications to mobile users [1]. Millions of mobile users interact with each other in MSNs, and MSNs will become one of the most important networking paradigms in future wireless mobile networks [2]. MSNs have always been trying to be innovative and leverage new technologies. The recently proposed integrated framework of computing, caching and communication (3C) can have positive impacts on MSNs. The integration of the 3C framework and MSNs can help create a new mechanism to share resources among users. The available resources that can be shared extend beyond information, for instance, the storage devices and computing capability contributed by users can also be shared. Additionally, the MSNs have an inherent advantage, i.e., the trust of mobile users is naturally formed through the social relationships and interactions. This makes the resource sharing between users more reliable. In this paper, we consider the specific 3C framework with *mobile edge computing* (MEC), *in-network caching*, and *device-to-device* (D2D) communications.

With MEC, computation resources are placed at the edge of wireless mobile networks in physical proximity to mobile users [3]. Compared to traditional mobile cloud computing, MEC can provide faster interactive response by low-latency connections. Therefore, MEC has been envisioned as a promising technique to offer agile and ubiquitous computation augmenting mobile services and applications, including social services and applications [4], [5]. Another promising technology is *in-network caching*, which can effectively reduce the duplicate content transmission in the network. Recent studies of applying the in-network caching technique in MSNs show that traffic loads and latency can be significantly reduced by caching contents in MSNs [2]. In addition, D2D communications can be beneficial to MSNs as well [6]. With D2D communications, users in close proximity can directly communicate with each other via D2D links, instead of accessing base stations (BSs) exclusively. When it comes to the content-centric MSNs, in spite of the smaller-sized storage (compared to that of BSs), the ubiquitous caching capability residing in mobile devices cannot be neglected due to their ubiquitous in-network distribution and ever-increasing storage size [6], [7].

Although some works have been done on applying recent advances of MEC, in-network caching and D2D to improve the performance of MSNs, the *knowledge of social relationships* among users in MSNs is largely ignored in these new paradigms to improve the reliability and efficiency of resource sharing and delivery in MSNs. In fact, social relationships have been investigated to enhance wireless networking. For example, Zhang et al. [8] exploit social ties to achieve a reliable D2D communication, which can improve packet transmission and reduce the workload on the network infrastructures. Pan

- *Y. He, C. Liang, and F.R. Yu are with the Department of Systems and Computer Engineering, Carleton University, Ottawa, ON K1S 5B6, Canada. E-mail: {heying, chengchaoliang}@sce.carleton.ca, Richard.Yu@Carleton.ca.*
- *Z. Han is with the University of Houston, Houston, TX 77004, and also with the Department of Computer Science and Engineering, Kyung Hee University, Seoul, South Korea. E-mail: zhan2@uh.edu.*

Manuscript received 19 Jan. 2018; revised 10 Apr. 2018; accepted 13 May 2018. Date of publication 13 Aug. 2018; date of current version 5 Mar. 2020. (Corresponding author: F. Richard Yu.)
Recommended for acceptance by J. Li.
Digital Object Identifier no. 10.1109/TNSE.2018.2865183

et al. [9] utilize social information as an essential element for designing forwarding algorithms in ad hoc networks. Social relationships also play an important role for routing in wireless environments [10].

Furthermore, the dynamic nature of the available resources has not been paid much attention in the existing literature. To fill in this gap, in this paper, we study trust-based social networks with recent advances of MEC, caching and D2D. Considering the integrated network, the allocation of resources for subscribed users is complicated, especially when the conditions of the network resources are varying with time. Therefore, we utilized a novel deep reinforcement learning approach to automatically achieve the resource allocation tasks.

In the following, we first review some of the existing excellent works that focus on efficient management and allocation schemes of computing, caching and communication resources. Some of the proposed schemes only consider one of these three types of resources. On the other hand, in some other works, two or three types of resources are jointly considered to effectively and efficiently achieve optimal performance metrics. Then, we discuss our contributions in this paper.

1.1 Related Works

With the increasing popularity of compute intensive applications, such as augmented reality, interative video games, etc., MEC is becoming a very promising paradigm that enables the possibility of offloading the computation tasks from resource-limited mobile devices to more powerful edge servers. MEC can bring various benefits, where minimizing energy consumption and minimizing latency are two major optimization objectives [11]. In the literature, various resource allocation approaches have been proposed to solve the formulated optimization problems. For example, Zhang et al. [12] propose an energy-efficient computation offloading scheme that jointly optimizes radio resource allocation and offloading to minimize the energy consumption of the offloading system with the latency constraint. In their scheme, the mobile devices are first classified into three types and then wireless channels are allocated to mobile devices based on their priority iteratively. By doing this, the optimization problem can be solved in polynomial complexity. Liu et al. [13] design an optimal computation task scheduling policy for MEC systems. They first analyze the average delay of each task and average power consumption at the mobile device side under the proposed scheduling policy using Markov chain theory. Then they formulate a delay minimization problem with the power constraint. An efficient one-dimensional search algorithm is used to derive the optimal offloading policy. In [14], the authors jointly optimize the radio resources and computational resources to minimize the total energy expenditure, where an iterative algorithm based on successive convex approximation technique is adopted to solve the formulated non-convex optimization problem. In [15], a distributed game theoretic approach is proposed to solve the NP-hard efficient computation offloading problem. Chen et al. [16] choose a model-free RL technique to solve the optimal traffic offloading strategy for heterogeneous cellular networks with dynamic traffic.

When caching technology is integrated into mobile networks, where to cache, what to cache, and how to cache are the major factors that make a significant influence on the system performance [11]. In [17], the authors investigate the proactive storage allocation problem over BSs in cellular networks, which is proved to be NP-hard. To get a low-complexity solution, a heuristic method is utilized and the convergence is proved by strict theoretic deduction. A caching-enabled D2D communication scheme is designed in [18], where the caching strategy is optimized by jointly considering users' social relationships and common interests with the constraint of hit ratio, delay and caching capacity. In fact, contents with the most popularity should be cached within limited caching capacity. Since the content popularity is varying with time, learning based caching policies are proposed in [19]. The authors formulate the distributed caching over small BSs (SBSs) as a reinforcement learning problem, and with the help of coded caching, the complicated caching problem is solved by being reduced to a convex optimization problem. In [20], Qiao et al. consider the mobility pattern of mobile users, and formulate the mobility-aware caching problem as an optimization problem to maximize the caching utility. The problem is solved by using a polynominal-time heuristic solution. He et al. exploit deep reinforcement learning to address the resource allocation problems in cache-enabled interference alignment networks [21], [22].

The comprehensive resource allocation schemes for efficient integration of computing, caching and communication resources to achieve the optimal performance have been developed, even though not yet been widely investigated. In [23], Zhou et al. design a novel information-centric heterogeneous network framework, and with the virtualization technology, communication, computing and caching resources are shared among users. They formulate the virtual resource allocation strategy as a joint optimization problem, which is a non-convex NP hard problem. An alternating direction method of multiplier (ADMM) approach is used to solve the problem by simplifying and relaxing the non-convex problem into a convex one. ADMM method is also used in [24] to achieve the optimal resource allocation strategy in wireless networks with MEC and in-network caching being jointly considered. He et al. propose an integrated framework that can enable dynamic orchestration of networking, caching and computing resources, where the dynamics are modeled as finite-state Markov chains [25], [26]. The authors use the deep Q-leaning approach to solve the complex problems with large numbers of system states and actions. Research on this topic of resource allocation strategies for integrated systems of computing, caching and communication will continue with the evolving of each of these three technologies.

1.2 Contributions

The main contributions of this article are summarized as follows.

- In this paper, we consider trust-based social networks specifically with MEC, in-network caching and D2D communications under the umbrella of a 3C framework. An optimization problem is formulated to maximize the network operator's utility with comprehensive considerations of trust values, computation capabilities, wireless channel qualities,

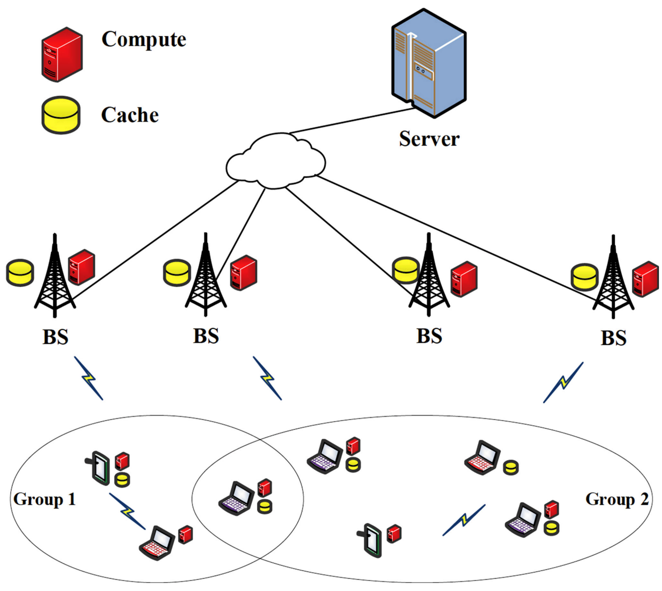


Fig. 1. A mobile social network with edge computing, caching and device-to-device (D2D) communications.

and the cache status of all the available BSs and D2D nodes.

- To be more realistic, we consider that the network conditions (i.e., trust value, computation capability, wireless channels, and cache status) are varying with time, and the dynamics of the computing capabilities, cache status and wireless channel conditions are modeled as Markov chains. In the meanwhile, the trust values are derived from both direct and indirect observations using Bayesian inference and Dempster-Shafer theory, respectively.
- The complexity of the integrated network is very high when we jointly consider the dynamic trust values, computation capabilities, wireless channel conditions and the cache status, and it is extremely difficult to solve the formulated optimization problem. In this paper, we exploit deep Q-learning based resource allocation strategy to solve the optimization problem without any explicit assumptions or simplifications.
- Google TensorFlow is used to implement the proposed deep Q-learning approach. Simulation results with different system parameters are presented to show the effectiveness of the proposed scheme. It is illustrated that the performance of MSNs can be significantly improved with the proposed approach.

The remainder of this paper is outlined as follows. We describe the system model in Section 2, which includes system description, network model, communication model, caching model and computing model. Next, the social trust scheme with uncertain reasoning is presented in Section 3. Section 4 formulates this system as a deep reinforcement learning problem. Simulation results are presented and discussed in Section 5. Finally, we give the conclusions and future work in Section 6.

2 System Model

In this section, we first present the system description. Next, the network model, communication model, cache model and computing model are presented, respectively.

2.1 System Description

MSNs have been developed rapidly to provide a variety of social services and applications to mobile users, focusing not only on the behaviors but also on the social needs of the users [1]. Compared to conventional mobile wireless networks, mobile users in MSNs do not always contact remote servers to request contents. Instead, mobile users in MSNs can directly obtain contents from each other within a community based on their social ties [8], [27].

In MSNs, huge amounts of information-rich data will be exchanged by mobile users. Although cloud computing is powerful, it is difficult for data centers to provide mobile users with low-latency services. To address these issues, MEC has been proposed to deploy computing resources closer to end users, which can efficiently improve the quality of service (QoS) for applications that require intensive computations and low latency [4], [28]. MEC applies the concept of cloud computing in network edge nodes, to facilitate services and applications, including mobile social services and applications.

In addition, due to user mobility and poor-quality wireless radio links, it is challenging to deliver huge amounts of data using the traditional client-server approach in MSNs [2]. Recent advances of in-network caching can be extended to MSNs to address this issue, which can efficiently reduce the duplicate content transmission in networks. This innovative content-centric approach has been studied in MSNs, which natively privilege the information (e.g., trusted information in a specific proximity of an event and a specific time period) rather than the node identity. In addition, with in-network caching, mobility and sporadic connectivity issues can be effectively addressed in MSNs.

Moreover, with D2D communications, users in close proximity can directly communicate with each other via D2D links, instead of accessing BSs exclusively. As a promising approach to offload traffic from BSs, D2D communications can enable the sharing of radio connectivity and direct information delivery between two close users [29], [30].

Security is always an important aspect in wireless applications and services [31], [32]. Trust-based security schemes are important detection-based approaches in MSNs. The definition of trust in MSNs is similar to that in sociology, where trust is interpreted as degrees of the belief that an entity will carry out tasks as expected [33]. In this paper, we present a framework for trust-based social networks specifically with MEC, in-network caching and D2D communications, which is depicted in Fig. 1. Under the framework, we illustrate a video content request task as an example, which will be elaborated in the following network model.

2.2 Network Model

In this paper, we consider the scenario with K BSs, and M mobile users that are considered as transmitters in the potential D2D communications. The network is operated by a central controller. Assume that there are L subscribed mobile users issuing video requests to the network operator, which is responsible for assigning a proper content provider to each requester, i.e., associating with one BS or setting up a one-to-one D2D communication. The sets are denoted as $\mathcal{C}_K = \{1, 2, ..., K\}$, $\mathcal{C}_M = \{1, 2, ..., M\}$ and $\mathcal{L} = \{1, 2, ..., L\}$, respectively. All the K BSs, the M D2D transmitters and the L mobile users are equipped with both caching and computing capabilities. The unicast D2D communication is available only when the mobile user's requested video is stored in the transmitter's cache. In addition, the transmitters can also provision mobile edge computing if they have spare computing capacity left at the considered moment.

In our system model, we assume that the lth mobile user, denoted as s_l, issues a video request. First, the network controller checks all the video providers, i.e., all the BSs and D2D transmitters. We denote $\mathcal{C} = \mathcal{C}_K \bigcup \mathcal{C}_M$ be the set of video providers, and let $c_i (1 \leq i \leq K)$ be the ith BS and $c_j (K+1 \leq j \leq K+M)$ be the jth transmitter. Each video provider's cache is labeled by a content indicator which indicates whether or not the requested video is being stored.

If the requested video is being stored at any of the video provider's caches, and the versions match up as well, the network controller will built up a communication between mobile user s_l and the optimal video provider. However, if all the stored videos' versions mismatch with the requested one, video transcoding should be performed at either mobile user s_l itself or at the video provider's side. On the other hand, If the requested video is not being stored at any of the caches, it has to associate with a proper BS.

Note that in this work, we use video version to represent video specification (e.g., H.263, H.264, MPEG2, or MPEG4). With the rapid growth of mobile services, increasing volumes of videos are played by mobile devices. Consequently, service providers often need to transcode the video contents into different specifications (e.g., bit rate, resolution, quality, etc.) with different QoS (e.g., delay) for heterogeneous mobile devices, networks, and user preferences.

2.3 Communication Model

Denote $h_{s_l}^{c_p}$ as the channel gain between mobile user s_l and content provider $c_p, \forall l \in \mathcal{L}, \forall p \in \mathcal{C}$. According to the Shannon theorem, the achievable rate of mobile user s_l when associating with provider c_p can be expressed as

$$r_{s_l}^{c_p} = B \log_2 \left(1 + \frac{p_T h_{s_l}^{c_p}}{N_o B} \right), \tag{1}$$

where B denotes the bandwidth allocated to each mobile user, p_T is the equally transmitted power, and N_o is the noise spectral density.

Here, we consider the realistic wireless channels, and actually $h_{s_l}^{c_p}$ is a continuous random variable. Such assumption is intractable for analysis [34], and therefore we model the wireless channels as the finite-state Markov channels (FSMC), which may achieve performance improvement compared with the traditional assumption of static channels. In our model, the channel gain $h_{s_l}^{c_p}$ is discretized and quantized into H levels: $\mathcal{H}_0$, if $h_0^* \leqslant h_{s_l}^{c_p} < h_1^*$; $\mathcal{H}_1$, if $h_1^* \leqslant h_{s_l}^{c_p} < h_2^*$; $\ldots$; $\mathcal{H}_{H-1}$, if $h_{s_l}^{c_p} \geqslant h_{H-1}^*$. The boundary values for a particular environment under certain criterion should be optimized for better performance [35]. However, for simplicity, uniformly setting the boundary values of FSMC is widely used in the literature (e.g., [35]). In this paper, all the boundary values from h_1^* to h_{H-1}^* are increasing in the same distance with each other. Each level corresponds to a state of the Markov chain, and therefore a H-element state space is formed. Due to the relationship between $h_{s_l}^{c_p}$ and $r_{s_l}^{c_p}$, for convenience we use $r_{s_l}^{c_p}$ to describe a state of the wireless channel. We consider the dynamic process, and the channel state realization of $r_{s_l}^{c_p}$ at time instant t is denoted as $\Upsilon_{s_l}^{c_p}(t)$. Based on a certain transitional probability, $\Upsilon_{s_l}^{c_p}(t)$ varies from one state to another as one time slot is gone. The transitional probability of $\Upsilon_{s_l}^{c_p}(t)$ from one state j_s to another state k_s is denoted as $\psi_{j_s k_s}(t)$. The $H \times H$ channel state transitional probability matrix between mobile user s_l and provider c_p is shown as:

$$\Psi_{s_l}^{c_p}(t) = \left[\psi_{j_s k_s}(t) \right]_{H \times H}, \tag{2}$$

where $\psi_{j_s k_s}(t) = \Pr\left(\Upsilon_{s_l}^{c_p}(t+1) = k_s \mid \Upsilon_{s_l}^{c_p}(t) = j_s \right)$.

2.4 Cache Model

Assume that there are totally I video contents in the network that mobile users can request for. The set of the contents is denoted as $\mathcal{I} = \{1, 2, ..., I\}$, which is ranked by popularity. Here, we consider finite cache capacity, i.e., each provider caches some of the I video contents, and refreshes the contents periodically. We consider that mobile user s_l has a video request $v_i, i \in \mathcal{I}$. After receiving the request message, the network controller checks every provider's content indicator for video v_i, i.e., $x_{c_p}^{v_i}, \forall p \in \mathcal{C}$. The content distribution indicator $x_{c_p}^{v_i} = 1$ means that video v_i is being stored in the cache of provider p; otherwise $x_{c_p}^{v_i} = 0$. Here, $x_{c_p}^{v_i}$ is viewed as a random variable, representing the state of the cache, and is modeled using a two-state (i.e., state 0 and 1) Markov chain [36]. The transitional probability matrix of the state $x_{c_p}^{v_i}$ is defined as:

$$\Gamma_{c_p}^{v_i}(t) = [\delta_{a_s b_s}(t)]_{2 \times 2}, \tag{3}$$

where $\delta_{a_s b_s}(t) = \Pr\left(x_{c_p}^{v_i}(t+1) = b_s \mid x_{c_p}^{v_i}(t) = a_s \right)$, and $a_s, b_s \in \{0, 1\}$.

When the cache capacity is assumed to be finite, the transitional probability matrix of the cache state can be derived based on different cache refreshment strategies. An important one is the least recently used (LRU) cache refreshment policy. The transitional probability matrix can be obtained using the following Markov chain flow matrix [36],

$$\Lambda_i = \begin{bmatrix} -\gamma_i & 0 & \cdots & 0 & 0 & \gamma_i \\ \zeta_i + \mu_i & -\beta - \mu_i & \cdots & 0 & 0 & \gamma_i \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mu_i & 0 & \vdots & \zeta_i & -\beta - \mu_i & \gamma_i \\ \mu_i & 0 & \vdots & 0 & \zeta_i & -\zeta_i - \mu_i \end{bmatrix}. \tag{4}$$

Here $\zeta_i = \beta - \gamma_i$, and γ_i represents the average request rate for the ith popular video v_i that can be denoted as

$$\gamma_i(t) = \frac{\beta}{\rho i^{\alpha}}, \tag{5}$$

where the request for v_i is assumed to arrive as a Poisson process with rate β [36], and the probability follows a Zipf-like distribution. Therefore, the probability of requesting content v_i is $1/\rho i^{\alpha}$, where $\rho = \sum_{i=1}^{I} 1/i^{\alpha}$ and α is the Zipf slope with $0 < \alpha < 1$. The video content is refreshed periodically, and the lifetime of any video content v_i follows an exponential distribution with mean $1/\mu_i$.

2.5 Computing Model

On the condition that the versions of the requested video at all the providers' caches do not match with the requested version, the associated provider has to extract the current video content and construct a computation task based on the input data information, as well as the number of CPU cycles. In the computation model, we construct the computation task as $Q_{s_l}^{v_i} = \{o_{s_l}^{v_i}, q_{s_l}^{v_i}\}$. The first parameter $o_{s_l}^{v_i}$ represents the size of the requested-version video, and the second parameter $q_{s_l}^{v_i}$ indicates the required number of CPU cycles that is needed to finish the computation task.

The computation capability of provider c_p assigned to mobile user s_l is denoted as $f_{s_l}^{c_p}$, which can be measured by the number of CPU cycles per second [15], [23]. In our network, multiple mobile users may associate with the same provider and share the computing device simultaneously, which would result in the fact that the computation capability for provider c_p is not exactly known at the next time instant. Therefore, the computation capability $f_{s_l}^{c_p}$ is modelled as a random variable, and equally divided into N discrete levels denoted by $\mathcal{E} = \{\mathcal{E}_0, \mathcal{E}_1, \ldots, \mathcal{E}_{N-1}\}$. The realization of the random variable $f_{s_l}^{c_p}$ is denoted as $F_{s_l}^{c_p}$ at time slot t. We model the transition of the computation capability level of a provider as a Markov process. The transitional probability matrix of $F_{s_l}^{c_p}$ can be expressed as:

$$\Theta_{s_l}^{c_p}(t) = \left[\iota_{x_s y_s}(t)\right]_{N\times N}, \tag{6}$$

where $\iota_{x_s y_s}(t) = \Pr\left(F_{s_l}^{c_p}(t+1) = y_s \,|\, F_{s_l}^{c_p}(t) = x_s\right)$, and $x_s, y_s \in \mathcal{E}$.

The consumed time for executing the task $Q_{s_l}^{v_i}$ at provider c_p can be obtained as $t_{s_l,c_p} = \frac{q_{s_l}^{v_i}}{F_{s_l}^{c_p}(t)}$. Furthermore, the rate of computation, i.e., the number of bits computed per second, can be given

$$r_{s_l,c_p}^{\text{comp}}(t) = a_{s_l,c_p}^{\text{comp}}(t)\frac{o_{s_l}^{v_i}}{t_{s_l,c_p}} = a_{s_l,c_p}^{\text{comp}}(t)\frac{f_{s_l}^{c_p}(t)o_{s_l}^{v_i}}{q_{s_l}^{c_p}}, \tag{7}$$

where $a_{s_l,c_p}^{\text{comp}}(t) \in \{0,1\}$ represents whether or not the computation task is decided to be performed at provider c_p.

3 Social Trust Scheme with Uncertain Reasoning

In this section, we will derive how to obtain the trust values of mobile users. We evaluate the trustworthiness of a mobile user by a real number Tr ranging from 0 to 1. In our model, the trust value Tr is jointly determined based on direct observations and indirect observations. The direct observation trust of a mobile user is defined as the estimated degree of trustworthiness from its directly-connected mobile users based on their past experiences. However, the subjective evaluation from direct connections may be prejudiced, and therefore in order to be more objective and impartial, we also consider the rating of trust from other indirectly-connected mobile users. Here, we denote the trust value from direct observations as Tr^D and the trust value from indirect observations as Tr^{InD}. By combining these two trust values, we can obtain a more accurately estimated trust value of a mobile user as

$$Tr = \omega Tr^D + (1-\omega)Tr^{InD}, \tag{8}$$

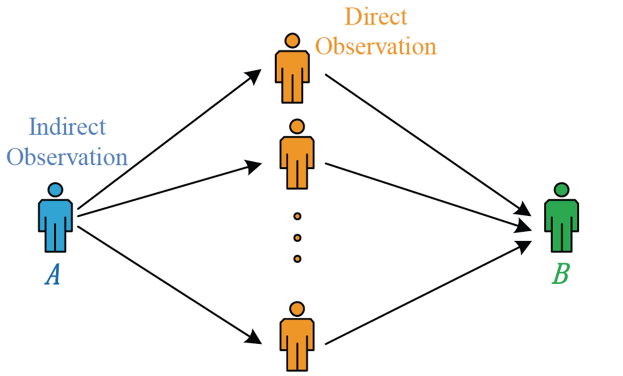


Fig. 2. Social trust evaluation with both direction observation and indirect observation.

where ω is the weight coefficient to adjust the weightiness between direct and indirect observations, and $0 \le \omega \le 1$.

The trust evaluation procedure in our model can be visually explained as Fig. 2. In the following, we will discuss how to obtain the trust evaluation from direct observations and indirect observations by using the Bayesian inference approach and Dempster-Shafer theory [37], respectively.

3.1 Trust Evaluation from Direct Observations

In the direct observations, consider that an observing mobile user can overhear the data forwarded by the observed mobile user, and identify the observed mobile user's malicious behaviors, such as discarding or modifying some of the original data.

Through multiple observations of the observed mobile user's behavior, the observing mobile user can evaluate the trust value by exploiting Bayesian inference [38], which is a method of statistical inference using the Bayes' theorem to update the probability for a hypothesis with more evidence becomes available.

Under the Bayesian framework, we model the trust of a mobile user as a continuous random variable, denoted as Φ, where ϕ takes values from 0 to 1. We assume that Φ follows a beta distribution [39], i.e., $\Phi \sim Beta(a,b)$, which is defined as follows with parameters a and b

$$Beta(a,b) = \frac{\phi^{a-1}(1-\phi)^{b-1}}{\int_0^1 \phi^{a-1}(1-\phi)^{b-1}\mathrm{d}\phi}, \tag{9}$$

for $0 \le \phi \le 1$. Here, since Φ is assumed to obey a beta distribution, the trust value can be represented by the two parameters a and b.

We summarize our belief of the trust Φ in a probability distribution iteratively as more observations are available. Assume that the prior probability density function (pdf) at the $(t-1)$th observation is known. Then, according to the Bayes theorem, the posterior distribution at the tth observation can be obtained with the pdf as

$$f_t(\phi) = \frac{f_t(x_t|\phi, y_t)f_{t-1}(\phi)}{\int_0^1 f_t(x_t|\phi, y_t)f_{t-1}(\phi)\mathrm{d}\phi}, \tag{10}$$

where x_t and y_t are the number of data packets that has been forwarded correctly and the number of packets received by the observed mobile user at the tth observation, respectively; $f_t(x_t|\phi, y_t)$ is the likelihood function, which follows a binomial distribution as

$$f_t(x_t|\phi, y_t) = \binom{y_t}{x_t}\phi^{x_t}(1-\phi)^{y_t-x_t}. \tag{11}$$

In Bayesian inference, the beta distribution is the conjugate prior probability distribution for the binomial distribution [38], [39]. Since the likelihood fuction $f_t(x_t|\phi, y_t)$ follows a binomial distribution, the priori distribution $f_{t-1}(\phi)$ is definitely assumed to follow a beta distribution, which reflects what is already known about the distribution of Φ at the $(t-1)$th observation. Given that the priori distribution $f_{t-1}(\phi)$ follows a beta distribution, the posterior distribution $f_t(\phi)$ also follows a beta distribution. Particularly, if $f_{t-1}(\phi) \sim Beta(a_{t-1}, b_{t-1})$ and x_t, y_t from the tth observation are also given, then

$$f_t(\phi) \sim Beta(a_{t-1} + x_t, b_{t-1} + y_t - x_t), t \geq 1. \tag{12}$$

At the very beginning of the observation, there is no evidence about the distribution of Φ, thus we assume that it follows a uniform distribution, i.e., $f_0(\phi) \sim Beta(1,1)$. Summarily, we can say that $f_t(\phi)$ follows $Beta(a_t, b_t)$ with parameters

$$\begin{aligned} a_t = a_{t-1} + x_t, \qquad & b_t = b_{t-1} + y_t - x_t, \\ a_0 = 1, \qquad & b_0 = 1. \end{aligned} \tag{13}$$

The trust value can be represented with the mathematical expectation of beta distribution as

$$\mathbb{E}_t[\Phi] = \frac{a_t}{a_t + b_t}. \tag{14}$$

Intuitively, the trust value of a mobile user is 0.5 at the beginning, and updated continuously by using the follow-up observations.

From the above discussion, we can see that past experiences play an important role in the Bayesian inference. In fact, recent behaviors of a mobile user should weight more in the trust evaluation. Here, we introduce a punishment factor for reputation fading, which gives more weights on misbehaviors in the Bayesian inference. The trust evaluation formula in (14) is revised as follows:

$$\mathbb{E}_t[\Phi] = \frac{a_t}{a_t + \tau b_t}, \tag{15}$$

where τ is the punishment factor, and $\tau \geq 1$. With the increment of τ, the trust value declines quickly.

The punishment factor makes the trust evaluation more realistic and reliable. First, if a mobile user once behaves maliciously, compared with those who have no bad records, its trust value will be lowered much more. Second, the trust value will not recover quickly even if he behaves well recently because of the constraint of the punishment factor. This helps distinguish the malicious mobile users quickly and prevent them disrupting others' trust evaluation. Based on the above deduction, the trust value from direct observation, Tr^D, is defined as:

$$Tr^D = \mathbb{E}_t[\Phi]. \tag{16}$$

3.2 Trust Evaluation from Indirect Observations

Apart from the direct observations, indirect observations from other mobile users are also very important in assessing the trustworthiness of an observed mobile user. Considering the indirect observations helps mitigate the situation that an observed mobile user is loyal to one mobile user but cheating on others. Assume that an observing mobile user collects observations from several other mobile users (also called subsidiary observing mobile user), and combines the collected evidence into a decision about the observed mobile user's trust value. However, these subsidiary observing mobile users may be untrustworthy or the evidence offered by them is unreliable.

The Dempster-Shafer theory can be used as an effective way to handle the uncertainty issue and combine the evidence from multiple subsidiary observers [40]. The core of this theory is based on two ideas: the degrees of belief about a proposition can be obtained from multiple subjective probabilities of a related theme, and these degrees of belief can be combined together under the condition that they are from independent evidence [37]. In the indirect observation, we assume that there are more than one subsidiary observing mobile users and the evidence provided by them is mutually independent.

3.2.1 Belief Function

As an introduction to the belief function, suppose that each subsidiary observing mobile user has two states, i.e., trustworthy and untrustworthy with probabilities p_1 and $1-p_1$ respectively. Assume that mobile user 1 claims that the observed mobile user B is trustworthy. If mobile user 1 is trustworthy, its statement is regarded as true; however, if mobile user 1 is untrustworthy, its statement is not necessarily untrue. We think that mobile user 1 provides p_1 degree of belief in the observed mobile user's trustworthiness (hypothesis H), 0 degree of belief in the untrustworthiness (hypothesis $\overline{H}$), and $1-p_1$ degree in the uncertainty, i.e., either trustworthy or untrustworthy (hypothesis U). Each hypothesis is assigned with a basic probability value $m(H)$, $m(\overline{H})$, $m(U)$ taking values from the interval $[0,1]$, respectively. In our scheme, we assign the basic probability value with the direct observation trust value. For example, if mobile user 1 believes that mobile user B is trustworthy, then the basic probability value for each possible hypothesis is:

$$\begin{aligned} m_1(H) &= T^D_{A1}, \\ m_1(\overline{H}) &= 0, \\ m_1(U) &= 1 - T^D_{A1}, \end{aligned} \tag{17}$$

where T^D_{A1} represents the trust value of mobile user 1 from the direct observation of mobile user B.

Oppositely, if mobile user 1 thinks that user B is untrustworthy, the basic probability value will be:

$$\begin{aligned} m_1(H) &= 0, \\ m_1(\overline{H}) &= T^D_{A1}, \\ m_1(U) &= 1 - T^D_{A1}. \end{aligned} \tag{18}$$

Concerning the formal definition of the belief function, let Ω be the universe of discourse, i.e., the set representing all possible states of the considered system, in this paper $\Omega = \{trustworthy, untrustworthy\}$. The power set 2^Ω refers to the set of all subsets of Ω, here $2^\Omega = \{\varnothing, \{trustworthy\}, \{untrustworthy\}, \Omega\}$. Any hypothesis A_i refers to a subset of 2^Ω, and is mapped into a basic probability $m(A_i)$ that reflects the proportion of total belief assigned to hypothesis A_i. These two conditions should be satisfied: $m(\varnothing) = 0$ and $\sum_{A_i \subseteq \Omega} m(A_i) = 1$. For any hypothesis B, the belief function is defined as

$$beli(B) = \sum_{A_i \subseteq B} m(A_i), \tag{19}$$

which represents the strength of the evidence that supports hypothesis B's provability.

3.2.2 Dempster's Rule of Combining Belief Functions

Based on the definition of belief function, the Dempster-Shafer theory combines multiple users' belief. Assuming that $beli_1(B)$ and $beli_2(B)$ are two belief functions over the same universe of discourse, Ω. the orthogonal sum of $beli_1(B)$ and $beli_2(B)$ can be defined as

$$\begin{aligned} beli(B) &= beli_1(B) \oplus beli_2(B) \\ &= \frac{\sum_{i,j,A_i \bigcap A_j = B} m_1(A_i) m_2(A_j)}{\sum_{i,j,A_i \bigcap A_j \neq \varnothing} m_1(A_i) m_2(A_j)}, \end{aligned} \tag{20}$$

where $A_i, A_j \subseteq \Omega$.

In our scenario, the combined degree of belief from mobile user 1 and mobile user 2 can be calculated as follows [37]:

$$\begin{aligned} m_1(H) \oplus m_2(H) =& \frac{1}{K} [m_1(H) m_2(H) + m_1(H) m_2(U) \\ &+ m_1(U) m_2(H)], \\ m_1(\overline{H}) \oplus m_2(\overline{H}) =& \frac{1}{K} [m_1(\overline{H}) m_2(\overline{H}) + m_1(\overline{H}) m_2(U) \\ &+ m_1(U) m_2(\overline{H})], \\ m_1(U) \oplus m_2(U) =& \frac{1}{K} m_1(U) m_2(U), \end{aligned} \tag{21}$$

where

$$\begin{aligned} K = & m_1(H) m_2(H) + m_1(H) m_2(U) + m_1(U) m_2(U) \\ &+ m_1(U) m_2(H) + m_1(U) m_2(\overline{H}) + m_1(\overline{H}) m_2(\overline{H}) \\ &+ m_1(\overline{H}) m_2(U). \end{aligned} \tag{22}$$

Following the rule of combination of belief, we can combine more degree of belief from other mobile users. Based on the Dempster-Shafer theory, T_{AB}^{N} is defined as:

$$Tr_{AB}^{InD} = m_1(H) \oplus m_2(H) \ldots \oplus m_n(H), \tag{23}$$

where there are totally between mobile user A and mobile user B.

4 Problem Formulation

In this section, we formulate an optimization problem of the integrated trust-based social network with the 3C framework with MEC, caching, and D2D communications. We assume that a mobile user requests for a video content to the integrated network. For the network operator, it should decide which BS or a D2D transmitter is assigned to serve the requesting user, whether or not the video transcoding (i.e., MEC) should be performed, and whether or not newly emerged contents should be cached. The network operator need a comprehensive consideration of many factors, including: the wireless channel conditions, whether or not the requested content is stored at the local cache, whether or not the content version is matched up, the computational capacity, the trustworthiness of a D2D transmitter. Here, we consider dynamic scenarios, i.e., the available network conditions are varying with time. We exploit deep Q-learning algorithm to solve the formulated optimization problem.

We first give a brief description about the deep Q-learning algorithm. In order to obtain the optimal policy, identifying the system's states, actions, and reward functions is necessarily required, which will be described in more details below.

4.1 Deep Q-Learning

Reinforcement learning is a category of machine learning, where an agent learns to take actions on the environment, and tries to obtain the most reward from the environment even though it faces with much uncertainty about the environment. The agent has to make a tradeoff between the exploration and exploitation, and adjusts its actions based on the delayed rewards. Usually, a reinforcement learning problem can be described by using a Markov Decision Process (MDP). More advantageously, reinforcement learning can handle the complex situations that the state, explicit transitional probability and immediate reward are not known [41], which is called model-free reinforcement learning.

Note: In order not to make confusions for readers from different backgrounds, here we briefly explain some related terms with reinforcement learning. The term dynamic programming (DP) refers to a collection of algorithms that can be utilized to compute optimal policies with the knowledge of a perfect model of the environment. However, classical DP algorithms are of limited applicability due to the assumption of a perfect model and the curse of dimensionality, which means the computational requirements grow exponentially with the number of system states [42]. To solve these problems, an alternative strategy that uses iterative algorithms to help estimate the value function is invented. This general strategy has been referred to as forward dynamic programming, iterative dynamic programming, adaptive dynamic programming, reinforcement learning, and neuro-dynamic programming [43]. The term that is being increasing adopted is approximate dynamic programming (ADP). However, artificial intelligence community continues to use "reinforcement learning", and in control theory community, the term "neuro-dynamic programming" is popularly used [44], [45].

Q-learning is one of the most widely used model-free reinforcement learning. Q-function is defined as the state-action function, and it can be obtained in a recursive manner using the available state, action, and reward information. In Q-learning, in order to obtain the best policy, it is necessarily required to estimate the Q-function. Sometimes the Q-function is implemented by a non-linear approximator, such as a neural network. However, a neural network achieves flexibility at the cost of stability [46]. Recently, a more advanced deep Q-learning is proposed, which approximate the Q-function with a deep neural network, and it is proven to be more stable and advantageous [46]. Compared with the traditional Q-learning, deep Q-learning has several outstanding improvements.

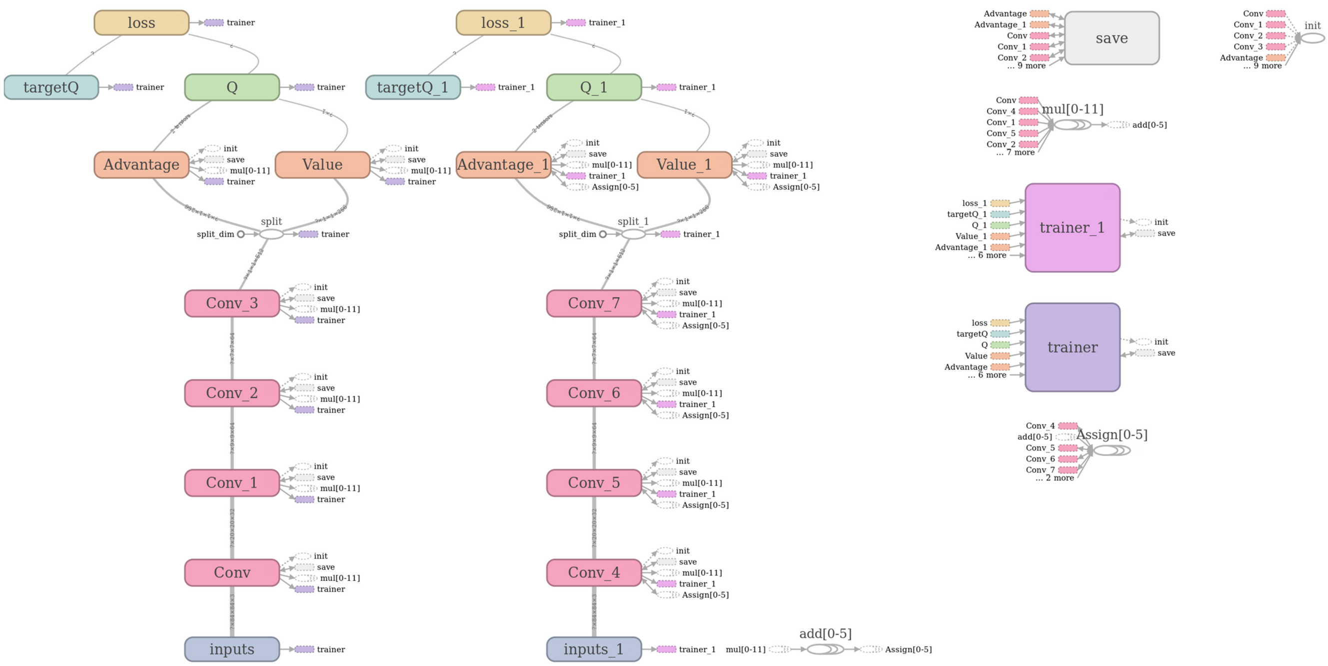


Fig. 3. Visualization of the proposed deep reinforcement learning algorithm using TensorBoard.

First of all, it goes from an one-layer fully-connected ordinary neural network to a multiple-layer sparsely-connected convolutional neural network. Convolutional neural networks, also called convolutional networks, play an important role in the evolution of deep learning, and they are some of the first deep models that can perform well [47]. Convolutional networks exploits convolution in substitute of matrix multiplication in at least one of the layers. For the convolution layer, the convolution operation is performed on the two arguments: *the input* and *the kernel*. The kernel is the representation for extracting a specific kind of feature (such as an edge, a contour, sharpening, an object identity, etc.), and it is also obtained through learning. The kernel is smaller than the input, and it slides over the input with a certain stride to produce *a feature map*. Different kernels produces different feature maps. Compared with the input, the size of the feature maps is significantly reduced, and these obtained feature maps are used as the input for the next layer. Convolution also makes it work that the input can be of variable size.

Another operation, called *p*ooling, appears in almost all the convolutional networks. Pooling, also referred as subsampling or downsampling, modifies the outputted feature map by only retaining some specific statistical information. Pooling includes several types, such as *the max pooling*, *the average pooling*, *the sum pooling*, *the* L^2 *norm pooling*, etc. In the case of max pooling, the largest element within a predefined rectangular neighborhood of the feature map will be taken as the representative output of that neighborhood. Pooling can be used as the next layer of convolution, and it can reduce the dimension of the feature map, and make the network invariant to small translations or distortions.

Deep Q-learning also utilizes a novel strategy called Experience Replay. It stores the past experiences into a replay memory, and randomly samples mini-batches from the pool to train the deep neural network, which refrains the agent from only concentrating on what the network is currently doing.

Additionally, deep Q-learning adopts two separate convolutional networks to generate the target Q values and the estimated Q values. Since deep Q-network is trained towards the target value by minimizing the loss function, using one network for both the estimated Q values and target Q values would lead to falling into feedback loops. For the sake of stabilizing the training process, the target deep Q-network's parameters is fixed and periodically updated.

In this paper, we use Minh et al.'s deep Q-network architecture [46] to train the learning process, but we decompose the Q-function, $Q(x,a)$, into two components, i.e., the value function $V(x)$, and the advantage function $A(x)$. Our deep Q-network separately computes the value and advantage functions, and combines them together into a single Q-function at the final fully-connected layer. This helps achieve a more robust estimate of the state value by decoupling it from specific actions. This architecture can be seen at the visualization of Fig. 3 in the simulation section.

4.2 System State

The system states for a subscriber s_l requesting video v_i at time slot $t \in \{0, 1, \ldots, T-1\}$ mainly includes five components: the realization $\Upsilon_{s_l}^{c_p}(t)$ of the random variables $\gamma_{s_l}^{c_p}$ (channel state), the realization $F_{s_l}^{c_p}(t)$ of the random variables $f_{s_l}^{c_p}$ (computation capability), the realization $X_{v_i}^{c_p}(t)$ of the random variables $x_{v_i}^{c_p}$ (content indicator), the version indicator $y_{v_i}^{c_p}$, and the trust value index $Tr_{s_l}^{c_p}$ for all the video providers, i.e., $\forall p \in \mathcal{C}$. Consequently, the system state vector can be described as follows.

$$\begin{aligned} \chi_{s_l}(t) = \Big[& \Upsilon_{s_l}^{c_1}(t), \Upsilon_{s_l}^{c_2}(t), \ldots, \Upsilon_{s_l}^{c_{K+M}}(t), \\ & F_{s_l}^{c_1}(t), F_{s_l}^{c_2}(t), \ldots, F_{s_l}^{c_{K+M}}(t), \\ & X_{v_i}^{c_1}(t), X_{v_i}^{c_2}(t), \ldots, X_{v_i}^{c_{K+M}}(t), \\ & y_{v_i}(t), \\ & Tr_{s_l}^{c_1}(t), Tr_{s_l}^{c_2}(t), \ldots, Tr_{s_l}^{c_{K+M}}(t) \Big]. \end{aligned} \tag{24}$$

Here, the trust index for all the BSs should be set to 1, i.e., $Tr^{c_p}_{s_l}(t) = 1, \forall c_p \in \mathcal{C}_K$; meanwhile, this index for all the D2D transmitters should be less than 1, i.e., $Tr^{c_p}_{s_l}(t) < 1, \forall c_p \in \mathcal{C}_M$. $y_{v_i}(t)$ denotes the indicator for whether or not the cached video version matches with the requested version. If yes $y_{v_i}(t) = 1$, if no $y_{v_i}(t) = 0$.

4.3 System Action

In our system, the network controller decides which video provider is assigned to the subscribed user, whether or not the computation offloading (video transcoding) should be performed, and whether or not the video provider should cache the new video. The network controller's composite action for subscriber s_l is given by

$$\mathbf{a}_{s_l}(t) = \{\mathbf{a}^{\text{comm}}_{s_l}(t), \mathbf{a}^{\text{comp}}_{s_l}(t), \mathbf{a}^{\text{cache}}_{s_l}(t)\}, \tag{25}$$

where the row vectors $\mathbf{a}^{\text{comm}}_{s_l}(t), \mathbf{a}^{\text{comp}}_{s_l}(t), \mathbf{a}^{\text{cache}}_{s_l}(t)$ are interpreted in more details as follows:

- The first row vector $\mathbf{a}^{\text{comm}}_{s_l}(t)$ is defined as

$$\mathbf{a}^{\text{comm}}_{s_l}(t) = [a^{\text{comm}}_{s_l,c_1}(t), a^{\text{comm}}_{s_l,c_2}(t), \ldots, a^{\text{comm}}_{s_l,c_{K+M}}(t)], \tag{26}$$

  where $a^{\text{comm}}_{s_l,c_p}(t)$ is the association indicator, and $a^{\text{comm}}_{s_l,c_p}(t) \in \{0,1\}$. If the subscribed user is associated with video provider c_p, $a^{\text{comm}}_{s_l,c_p}(t) = 1$, otherwise $a^{\text{comm}}_{s_l,c_p}(t) = 0$.
- The second row vector $\mathbf{a}^{\text{comp}}_{s_l}(t)$ is defined as

$$\mathbf{a}^{\text{comp}}_{s_l}(t) = [a^{\text{comp}}_{s_l,c_1}(t), a^{\text{comp}}_{s_l,c_2}(t), \ldots, a^{\text{comp}}_{s_l,c_{K+M}}(t)], \tag{27}$$

  where $a^{\text{comp}}_{s_l,c_p}(t)$ indicates the computation offloading decision, and $a^{\text{comp}}_{s_l,c_p}(t) \in \{0,1\}$. If the network controller has decided that the computation task should be performed at the provider c_p's device, $a^{\text{comp}}_{s_l,c_p}(t) = 1$; on the other hand, if the computation is decided to be executed on subscriber's own mobile device, $a^{\text{comp}}_{s_l,c_p}(t) = 0$.
- The third row vector $\mathbf{a}^{\text{cache}}_{s_l}(t)$ is defined as

$$\mathbf{a}^{\text{cache}}_{s_l}(t) = [a^{\text{cache}}_{s_l,c_1}(t), a^{\text{cache}}_{s_l,c_2}(t), \ldots, a^{\text{cache}}_{s_l,c_{K+M}}(t)], \tag{28}$$

  where $a^{\text{cache}}_{s_l,c_p}(t)$ means the whether or not the video provider c_p should cache the newly-emerged video or the new version, and $a^{\text{cache}}_{s_l,c_p}(t) \in \{0,1\}$. If the network controlled decides that c_p caches the video, $a^{\text{cache}}_{s_l,c_p}(t) = 1$, otherwise $a^{\text{cache}}_{s_l,c_p}(t) = 0$.

4.4 Reward Function

In this paper, we set the system reward to be the total revenue (i.e., utility) of the network operator. The network operator charges the subscribed user s_l for associating with the video provider, which is denoted as λ_{s_l} unit price per bps. On the condition that video transcoding (computation offloading) is decided to be executed on video provider's side, the operator can charge for its computing service, which is defined as ν_{s_l} unit price per bps. In addition, if the network controller decides to let the BSs or D2D transmitters cache the new video or new version, the operator gets a potential revenue on estimated backhaul save, which is denoted as κ_{s_l} unit price per Hz.

On the other hand, the operator has to pay for the rented spectrum and backhaul bandwidth. The unit price for the usage of spectrum is defined as δ_{c_p} per Hz for provider c_p. Moreover, if video transcoding is performed, a certain amount of energy will be consumed for the computing, the network operator is obliged to pay η_{c_p} unit price for per consumed Joule. The cost of caching the video content in the memory of provider c_p is denoted as ς_{c_p} unit price per unit space.

The system reward for serving subscribed user s_l requesting video v_i is defined as the network operator's total revenue, and it is formulated as a function of the system states, and actions. The system actions determine whether or not the reward can be optimized. Here, we define the reward function for a specific subscribed user s_l as:

$$\begin{aligned}
R^{v_i}_{s_l}(t) &= \sum_{c_p \in \mathcal{C}} \Big[R^{\text{comm}}_{s_l,c_p}(t) + R^{\text{comp}}_{s_l,c_p}(t) + R^{\text{cache}}_{s_l,c_p}(t)\Big] \\
&= \sum_{c_p \in \mathcal{C}} a^{\text{comm}}_{s_l,c_p}(t)\big(\lambda_{s_l} Tr^{c_p}_{s_l}(t) \Upsilon^{c_p}_{s_l}(t)(1 - w^{\text{comp}} a^{\text{comp}}_{s_l,c_p}(t)) - \delta_{c_p} B \\
&\quad - (1 - X^{c_p}_{v_i}(t))\sigma^{c_p} \Upsilon^{c_p}_{s_l}(t)\big) \\
&\quad + \sum_{c_p \in \mathcal{C}} (1 - y^{c_p}_{v_i}) a^{\text{comp}}_{s_l,c_p}(t)(\nu_{s_l} Tr^{c_p}_{s_l}(t) \frac{F^{c_p}_{s_l}(t) o^{v_i}_{s_l}}{q^{v_i}_{s_l}} - \eta_{c_p} q^{c_p}_{s_l} e_{c_p}) \\
&\quad + \sum_{c_p \in \mathcal{C}} a^{\text{cache}}_{s_l,c_p}(t)(\kappa_{s_l} Tr^{c_p}_{s_l}(t) \Upsilon^{c_p}_{s_l}(t) - \varsigma_{c_p} o^{v_i}_{s_l}).
\end{aligned} \tag{29}$$

The above reward function is composed of three terms, i.e., the earnings from communications, computing and caching, respectively. Here, the trust index $Tr^{c_p}_{s_l}(t)$ is added to each incoming revenue term. For the first communication earnings, $\lambda_{s_l} Tr^{c_p}_{s_l}(t)\Upsilon^{c_p}_{s_l}(t)(1 - w^{\text{comp}} a^{\text{comp}}_{s_l,c_p}(t))$ denotes the available earnings for providing the video transmitting service to subscriber s_l. Note that if video transcoding is decided to be performed, the video will be compressed and the corresponding income will be reduced. $\delta_{c_p} B$ is the cost for consuming the radio spectrum bandwidth. $(1 - X^{c_p}_{v_i}(t))\sigma^{c_p}\Upsilon^{c_p}_{s_l}(t)$ is the cost for possible consumed backhaul bandwidth: if $X^{c_p}_{v_i}(t) = 1$, i.e., the requested video exists in the cache of provider c_p, no backhual cost is needed; otherwise $X^{c_p}_{v_i}(t) = 0$, the video has to be fetched from the Internet and backhaul cost is unavoidable. Note that σ^{c_p} is a very important parameter. It is not realistic to build up a D2D communication when the requested video does not exist in its cache. Thus, σ^{c_p} should be set to be an extremely large number for $c_p, \forall c_p \in \mathcal{C}_M$. For the second computing earnings, $(1 - y^{c_p}_{v_i})$ is used to indicate whether or not the version is matching. $\nu_{s_l} Tr_{s_l}(t)^{c_p} \frac{F^{c_p}_{s_l}(t) o^{v_i}_{s_l}}{q^{v_i}_{s_l}}$ represents the gaining for implementing video transcoding, and $\eta_{c_p} q^{c_p}_{s_l} e_{c_p}$ is the expenditure for the energy consumption, where e_{c_p} denotes consumed energy for running one CPU cycle. For the third caching earnings, $\kappa_{s_l} Tr_{s_l}(t)^{c_p} \Upsilon^{c_p}_{s_l}(t)$ is the estimated future income of the saved backhaul bandwidth if the new video or new version is decided to be cached. Here the saved backhaul bandwidth is equal to the wireless communication rate. At last, $\varsigma_{c_p} o^{v_i}_{s_l}$ denotes the cost for caching the content.

$R^{v_i}_{s_l}(t)$ is the system's immediate reward, i.e., the network operator gets $R^{v_i}_{s_l}(t)$ at state $\chi_{s_l}(t)$ when action $\mathbf{a}_{s_l}(t)$ is performed in time slot t. However, a maximum immediate value is not a guarantee for the maximum long-term future rewards. Thus, we should also consider a big picture. A future reward with a discount factor φ is reasonable, which can be denoted as

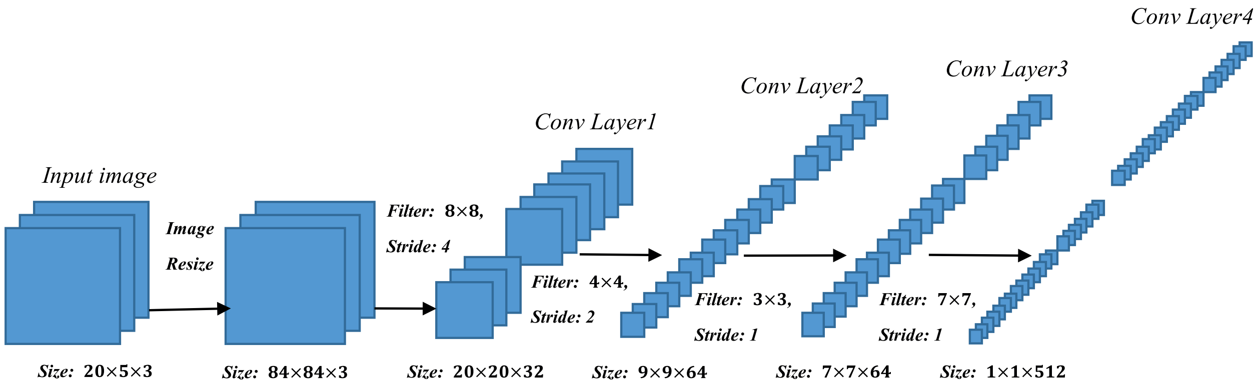


Fig. 4. The training process of the deep convolution networks.

$$R^f_{s_l,v_i} = \max_{\pi} \mathbb{E}\left[\sum_{t=0}^{t=T-1} \varphi^t R^{v_i}_{s_l}(t)\right], \tag{30}$$

where π represents a Q-learning policy, i.e., a series of actions where a specific action will be executed given a specific system state, and φ^t approximates to be zero when T is large enough. In fact, a condition for terminating the training process should be set.

The objective of adopting deep Q-learning into our network model is to help find an optimization policy that can maximize the cumulated future rewards for the network operator.

5 Simulation Results and Discussions

In this section, we evaluate the performance of proposed scheme using computer simulations. We use TensorFlow [48] in our simulations to implement deep Q-learning. For performance comparison, the following four algorithms are presented: 1). Proposed scheme, which considers MEC, caching, D2D, as well as both direct observation and indirect observation; 2). Proposed scheme w.o. indirect observation, which does not consider indirect observation; 3). An existing scheme w.o. mobile edge computing [49]; and 4). An existing scheme w.o. D2D communications [2].

5.1 Simulation Settings

In the simulations, we consider an MSN consisting of 5 BSs, and 15 D2D transmitters. The radius of the cell is set to 500 m. Because D2D communications typically perform within short ranges, we use a clustered-based distribution model [50], where multiple users are located within one cluster with a radius of 50m. In the network, there are 20 subchannels, each of which has a bandwidth of 180 KHz. The transmit powers of D2D transmitter and BS are 24dBm and 46 dBm, respectively. The noise spectral density is -174 dBm/Hz. A loss model $35.3 + 37.6 \log(d(m))$ [51] is used. In addition, block fading with a block size of 100 is assumed for channel fading. Moreover, we assume that there are totally 10 types of contents distributed in the network, and each content cache state follows the Markov model. We set the cache state transition probability of staying in the same state in the BS as 0.6 (0.3 in the D2D transmitter), and set the transition probability from one state to another as 0.4 (0.7 in the D2D transmitter). We further assume a Zipf popularity distribution in the simulations, with $\theta = 1.5$ [52]. The computation states of MEC servers follow the Markov model. We assume that the computation state of the MEC server can be very low, low, medium, high, and very high.

We assume that there are two types of D2D transmitters in the network: normal nodes, which share the content and perform computing normally, and compromised nodes, which modify contents maliciously. The BSs are assumed to be not compromised due to the physical security of BSs. We also assume that the number of compromised nodes is much less than the total number of nodes in the network. The attackers are independent. Hence, there is no collusion attack in the network.

In our simulations, we used a GPU-based server, which has 4 Nvidia GPUs with version GTX TITAN. The CPU is Intel Xeno E5-2683 v3 with 128G memory. The software environment we utilized is TensorFlow 0.12.1 with Python 2.7 on Ubuntu 14.04 LTS.

The visualization of our deep Q-network architecture is presented in Fig. 3 using TensorFlow's build-in module, TensorBoard. From Fig. 3, it can be clearly observed that double deep convolutional networks are used, each network has four convolutional layers, and the value functions and advantage functions are separately computed. The models can be saved and loaded to continue the training or testing process. Specifically, the training process of the deep convolutional networks are shown in Fig. 4. We formulate the 5 BSs and 15 D2D transmitters as the rows, and the 5 levels of computational capacity as the columns in a grid image. Whether or not the required content is in the local cache is characterized by different colors in each small grid. The initial input image of size $20 \times 5 \times 3$ is first resized into $84 \times 84 \times 3$. Through 4 layers of convolutional operations, the output is 512 nodes, and these nodes will be fully connected to be trained in the deep reinforcement learning.

5.2 Simulation Results

The convergence performance of the proposed scheme using the deep reinforcement learning algorithm is shown in Fig. 5. We can observe that, at the beginning of the learning process, the total utility of the proposed scheme is very low. As the number of the episodes increases, the total utility increases until it converges to a relatively stable value, which is around 5900. From Fig. 5, we can also observe the convergence performance of the proposed scheme with different learning rates. The learning rate has effects on the convergence performance. For example, the convergence of the proposed scheme is faster when the learning rate is 0.001 compared to the cases when the learning rates are

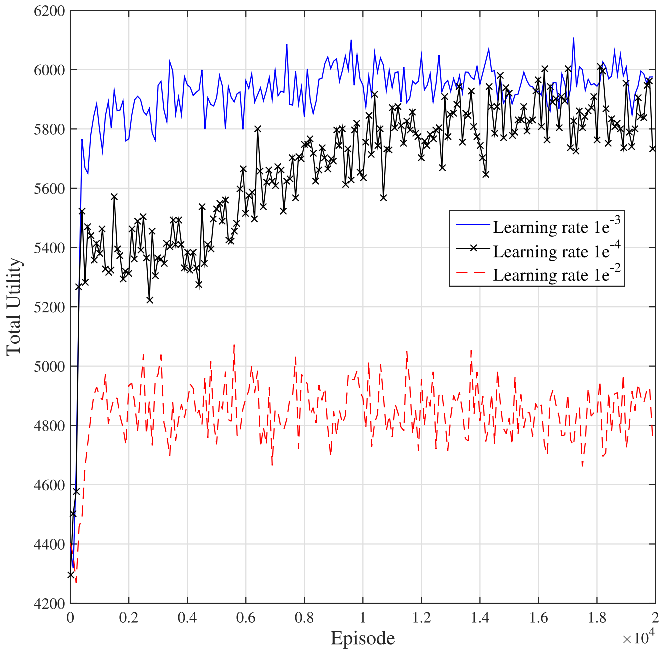


Fig. 5. Convergence performance with different learning rates.

0.0001. However, this does not mean that a larger learning rate is always better, because a larger learning rate may result in local optimum in the problem instead of the global optimum. For example, in Fig. 5, with the learning rate of 0.01, the algorithm converges to the total utility of around 4900, which is much lower than the converged values achieved by other learning rates. Therefore, it is important to choose an appropriate learning rate for a specific problem. For our studied problem, we choose the learning rate of 0.001 in the rest of the simulations.

The effects of average size per content on the total reduced backhaul usage is shown in Fig. 6. There two malicious D2D transmitters, and there are 4 types of contents in the system. We can see from Fig. 6 that the total reduced backhaul usage increases with the increase of the

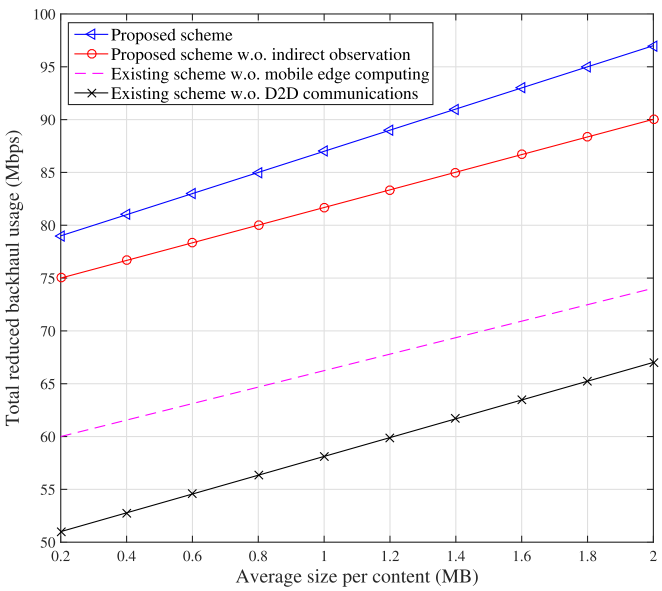


Fig. 6. The total reduced backhaul usage versus average size per content.

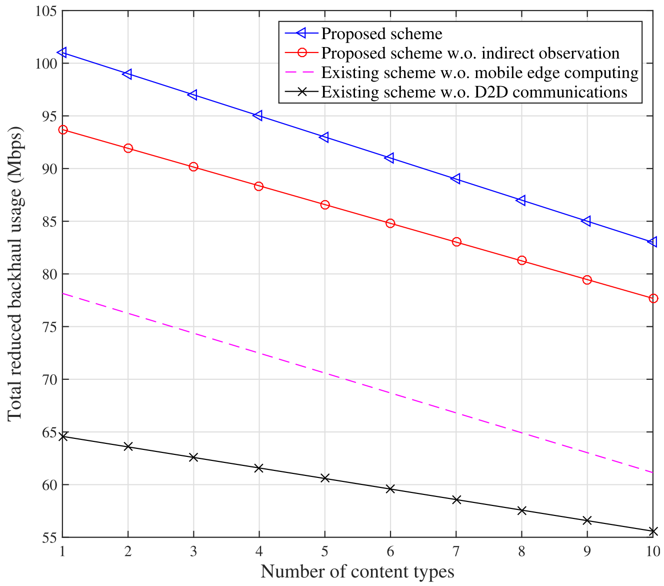


Fig. 7. The total reduced backhaul usage versus the number of content types.

average size per content. This is because a larger content size will increase the fee for caching the content, which induces a lower gain of caching utility, and thus the system is reluctant to cache the contents in the network. Compared to the existing schemes without mobile edge computing and D2D communications, the proposed scheme has larger total reduced backhaul usage due to the benefits of mobile edge computing and D2D communications. In addition, without indirect observation, the accuracy of trust evaluation is lower in the proposed scheme, which induces a lower gain of mobile edge computing and caching, and lower total reduced backhaul usage.

Figs. 7 and 8 show the effects of the number of content types on the total reduced backhaul usage and the total utility, respectively. In these simulations, the average size per content is 1.8 MB, and there are two malicious D2D

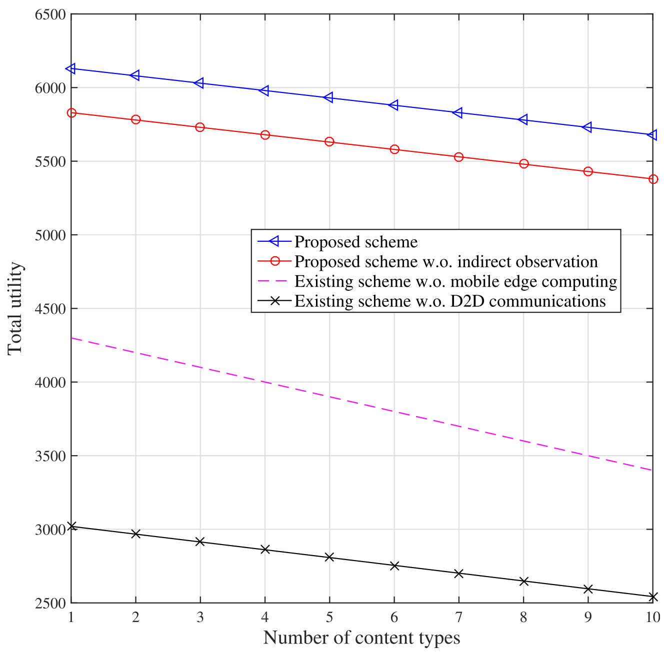


Fig. 8. The total utility versus the number of content types.

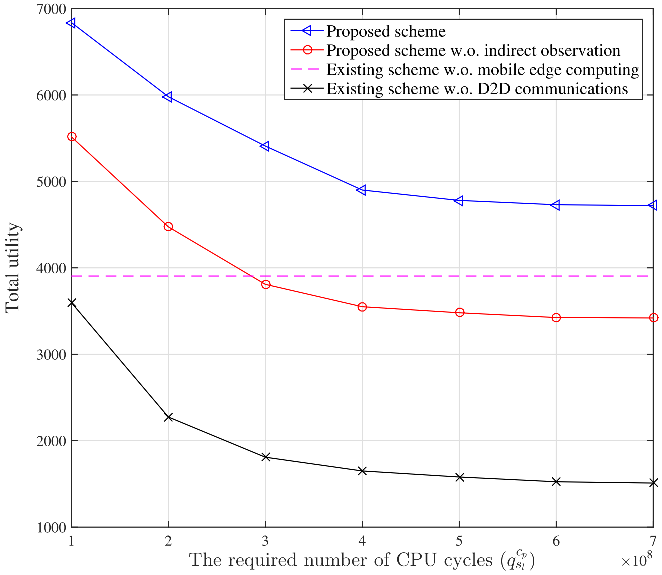


Fig. 9. The total utility versus the required number of CPU cycles for video transcoding.

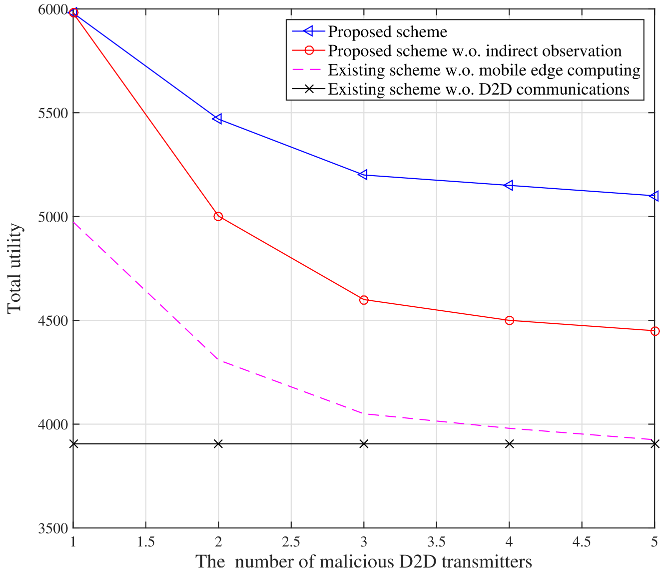


Fig. 11. The total utility versus the number of malicious D2D transmitters.

transmitters. We can see from both figures that the reduced backhaul usage and the total utility decrease as the number of content types increases. This is because more content types in the system will lead to the scenarios that a requesting mobile user is unlikely to find the specific content from the BSs and D2D transmitters due to the limited storage. In addition, more content types will result in reduced the popularity of all contents according to the Zipf popularity distribution, thus decreasing caching gain.

Fig. 9 shows the effects of the required numbers of CPU cycles for video transcoding. From Fig. 9, we can see that the required number of CPU cycles has no effects on the total utility of the existing scheme without mobile edge computing. This is because the total utility will not be changed with different required numbers of CPU cycles if mobile edge computing is not available in the system. In the proposed scheme and the existing scheme without D2D communications, the total utility decreases exponentially with the increase of the required number of CPU cycles for video transcoding. This is because a larger number of required number of CPU cycles will result in a higher consumed computation energy, and consequently a lower gain of computation utility. Therefore, the total utility decreases with the required number of CPU cycles for video transcoding.

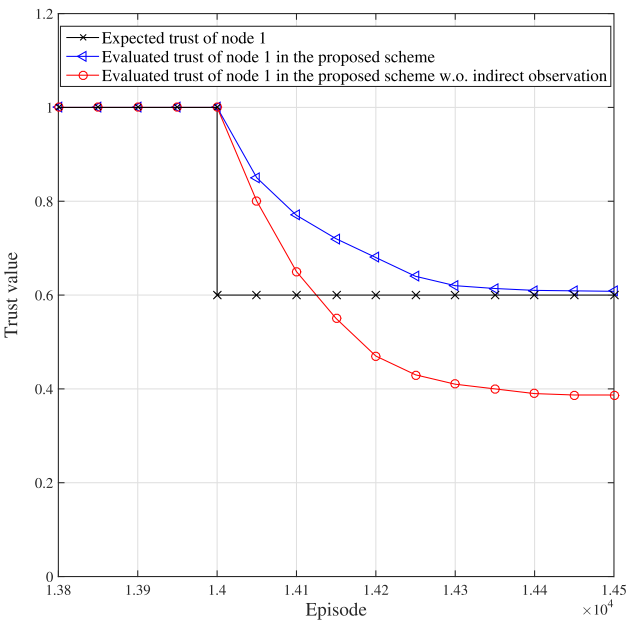


Fig. 10. The trust value versus the episodes.

Next, we study the performance of the proposed social trust scheme. Assume that, at episode 14K, D2D transmitter 1 changes its maliciousness to 0.6. Our goal is to observe the accuracy of trust tracking in this scenario where the malicious behavior of a D2D transmitter changes rapidly. Fig. 10 shows the trust tracking of the system using direct observation with Bayesian inference and the proposed scheme using both direct observation with Bayesian inference and indirect observation with the Dempster-Shafer theory. We can observe from Fig. 10 that only direct observation can result in inaccurate trust values. By contrast, the proposed scheme can track the trust value accurately with both direct observation and indirect observation.

The number of malicious D2D transmitters in the network also has a significant impact on the performance the network. Here, We investigate the system utility with the number malicious D2D transmitters, from 1 to 5, in a 15 D2D transmitter environment. The basic parameter is the same as above. Fig. 11 shows that, as the number of malicious D2D transmitters increases, the utility drops dramatically. When the number of malicious D2D transmitters reaches to one third of the total number of D2D transmitters in the network, the utility decreases to about half of the utility in the network with 1 malicious nodes. From this figure, we can see that the network is deeply affected by the number of malicious D2D transmitters.

6 Conclusions and Future Work

MSNs have become one of the most important networking paradigms in future wireless mobile networks. Recent advances of wireless mobile networks can have significant impacts on the performance of MSNs. In this paper, we studied recent advances in mobile edge computing, in-network caching and D2D communications in MSNs. In addition, we considered the knowledge of social relationships in these new paradigms to improve the security and efficiency of MSNs. Specifically, we presented a social trust scheme with both direct observation using Bayesian inference and indirect observation using the Dempster-Shafer theory. We further proposed a deep Q-learning approach to study this complicated system. Extensive simulation results were presented to show the effectiveness of the proposed scheme. Since energy efficiency is becoming increasingly important in the big data era [53], [54] Future work is in progress to consider energy-efficient resource allocation strategy in the proposed integrated framework.

Acknowledgments

The research is partially supported by Natural Sciences and Engineering Research Council of Canada (NSERC), US MURI, NSF CNS-1717454, CNS-1731424, CNS-1702850, CNS-1646607,ECCS-1547201. We thank the reviewers for their detailed reviews and constructive comments, which have helped to greatly improve the quality of this paper.

References

[1] N. Vastardis and K. Yang, "Mobile social networks: Architectures, social properties, and key research challenges," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1355–1371, Jul.-Sep. 2013.

[2] Z. Su and Q. Xu, "Content distribution over content centric mobile social networks in 5G," *IEEE Commun. Mag.*, vol. 53, no. 6, pp. 66–72, Jun. 2015.

[3] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, A. Neal, et al., ETSI, "Mobile-edge computing introductory technical white paper," *White Paper, Mobile-edge Computing (MEC) industry initiative*, Sep. 2014.

[4] N. Kumar, S. Zeadally, and J. J. P. C. Rodrigues, "Vehicular delay-tolerant networks for smart grid data management using mobile edge computing," *IEEE Comm. Mag.*, vol. 54, no. 10, pp. 60–66, Oct. 2016.

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

[6] Y. Meng, C. Jiang, H. H. Chen, and Y. Ren, "Cooperative device-to-device communications: Social networking perspectives," *IEEE Netw.*, vol. 31, no. 3, pp. 38–44, May 2017.

[7] K. Wang, F. R. Yu, and H. Li, "Information-centric virtualized cellular networks with device-to-device (D2D) communications," *IEEE Trans. Veh. Tech.*, vol. 65, no. 11, pp. 9319–9329, Nov. 2016.

[8] Y. Zhang, E. Pan, L. Song, W. Saad, Z. Dawy, and Z. Han, "Social network aware device-to-device communication in wireless networks," *IEEE Trans. Wireless Commun.*, vol. 14, no. 1, pp. 177–190, Jan. 2015.

[9] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: Social-based forwarding in delay-tolerant networks," *IEEE Trans. Mobile Comput.*, vol. 10, no. 11, pp. 1576–1589, Nov. 2011.

[10] E. Bulut and B. K. Szymanski, "Friendship based routing in delay tolerant mobile social networks," in *Proc. IEEE GLOBECOM*, Dec. 2010, pp. 1–5.

[11] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A survey on mobile edge networks: Convergence of computing, caching and communications," *IEEE Access*, vol. 5, pp. 6757–6779, Mar. 2017.

[12] K. Zhang, Y. Mao, S. Leng, Q. Zhao, L. Li, X. Peng, L. Pan, S. Maharjan, and Y. Zhang, "Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks," *IEEE Access*, vol. 4, pp. 5896–5907, Aug. 2016.

[13] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2016, pp. 1451–1455.

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

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

[16] X. Chen, J. Wu, Y. Cai, H. Zhang, and T. Chen, "Energy-efficiency oriented traffic offloading in wireless networks: A brief survey and a learning approach for heterogeneous cellular networks," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 4, pp. 627–640, Jan. 2015.

[17] J. Gu, W. Wang, A. Huang, and H. Shan, "Proactive storage at caching-enable base stations in cellular networks," in *Proc. IEEE Int. Symp. Personal Indoor Mobile Radio Commun.*, Sep. 2013, pp. 1543–1547.

[18] B. Bai, L. Wang, Z. Han, W. Chen, and T. Svensson, "Caching based socially-aware D2D communications in wireless content delivery networks: A hypergraph framework," *IEEE Wireless Commun.*, vol. 23, no. 4, pp. 74–81, Aug. 2016.

[19] A. Sengupta, S. Amuru, R. Tandon, R. M. Buehrer, and T. C. Clancy, "Learning distributed caching strategies in small cell networks," in *Proc. 11th Int. Symp. Wireless Commun. Syst.*, Aug. 2014, pp. 917–921.

[20] Y. Guan, Y. Xiao, H. Feng, C.-C. Shen, and L. J. Cimini, "Mobicacher: Mobility-aware content caching in small-cell networks," in *Proc. IEEE GLOBECOM*, Feb. 2014, pp. 4537–4542.

[21] Y. He, C. Liang, F. R. Yu, N. Zhao, and H. Yin, "Optimization of cache-enabled opportunistic interference alignment wireless networks: A big data deep reinforcement learning approach," in *Proc. IEEE Int. Conf. Commun.*, May 2017, pp. 1–6.

[22] Y. He, Z. Zhang, F. R. Yu, N. Zhao, H. Yin, V. C. Leung, and Y. Zhang, "Deep-reinforcement-learning-based optimization for cache-enabled opportunistic interference alignment wireless networks," *IEEE Trans. Veh. Tech.*, vol. 66, no. 11, pp. 10 433–10 445, Nov. 2017.

[23] Y. Zhou, F. R. Yu, J. Chen, and Y. Kuo, "Resource allocation for information-centric virtualized heterogeneous networks with in-network caching and mobile edge computing," *IEEE Trans. Veh. Tech.*, vol. 66, no. 12, pp. 11 339–11 351, Dec. 2017.

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

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

[26] Y. He, F. R. Yu, N. Zhao, V. C. Leung, and H. Yin, "Software-defined networks with mobile edge computing and caching for smart cities: A big data deep reinforcement learning approach," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 31–37, Dec. 2017.

[27] Z. Su and Q. Xu, "Security-aware resource allocation for mobile social big data: A matching-coalitional game solution," *IEEE Trans. Big Data*, to be published, doi: 10.1109/TBDATA.2017.2700318.

[28] X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, and S. Chen, "Vehicular fog computing: A viewpoint of vehicles as the infrastructures," *IEEE Trans. Veh. Tech.*, vol. 65, no. 6, pp. 3860–3873, Jun. 2016.

[29] A. Asadi, Q. Wang, and V. Mancuso, "A survey on device-to-device communication in cellular networks," *IEEE Commun. Surveys Tutorials*, vol. 16, no. 4, pp. 1801–1819, Oct.-Dec. 2014.

[30] C. Yang, J. Li, P. Semasinghe, E. Hossain, S. M. Perlaza, and Z. Han, "Distributed interference and energy-aware power control for ultra-dense D2D networks: A mean field game," *IEEE Trans. Wireless Comm.*, vol. 16, no. 2, pp. 1205–1217, Feb. 2017.

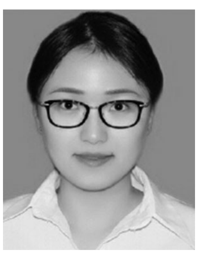
[31] K. Hamedani, L. Liu, R. Atat, J. Wu, and Y. Yi, "Reservoir computing meets smart grids: attack detection using delayed feedback networks," *IEEE Trans. Ind. Inf.*, vol. 14, no. 2, pp. 734–743, Feb. 2018.

[32] R. Atat, L. Liu, H. Chen, J. Wu, H. Li, and Y. Yi, "Enabling cyber-physical communication in 5G cellular networks: challenges, spatial spectrum sensing, and cyber-security," *IET Cyber-Phys. Syst. Theory Appl.*, vol. 2, no. 1, pp. 49–54, Apr. 2017.

[33] I. R. Chen, F. Bao, and J. Guo, "Trust-based service management for social Internet of Things systems," *IEEE Trans. Dependable Secure Comput.*, vol. 13, no. 6, pp. 684–696, Nov. 2016.

[34] F. Parzysz, M. Di Renzo, and C. Verikoukis, "Power-availability-aware cell association for energy-harvesting small-cell base stations," *IEEE Trans. Wireless Commun.*, vol. 16, no. 4, pp. 2409–2422, Apr. 2017.

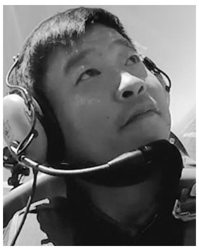
[35] H. S. Wang and N. Moayeri, "Finite-state Markov channel-a useful model for radio communication channels," *IEEE Trans. Veh. Tech.*, vol. 44, no. 1, pp. 163–171, Feb. 1995.
[36] H. Gomaa, G. G. Messier, C. Williamson, and R. Davies, "Estimating instantaneous cache hit ratio using Markov chain analysis," *IEEE/ACM Trans. Netw.*, vol. 21, no. 5, pp. 1472–1483, Oct. 2013.
[37] T. M. Chen and V. Venkataramanan, "Dempster-Shafer theory for intrusion detection in ad hoc networks," *IEEE Internet Comput.*, vol. 9, no. 6, pp. 35–41, Nov. 2005.
[38] R. Changiz, H. Halabian, F. R. Yu, I. Lambadaris, H. Tang, and C. M. Peter, "Trust establishment in cooperative wireless networks," in *Proc. IEEE MILCOM*, Nov. 2010, pp. 1074–1079.
[39] C. Zouridaki, B. L. Mark, M. Hejmo, and R. K. Thomas, "Hermes: A quantitative trust establishment framework for reliable data packet delivery in MANETs," *J. Comput. Secur.*, vol. 15, no. 1, pp. 3–38, Jan. 2007.
[40] S. Bu, F. R. Yu, P. Liu, P. Manson, and H. Tang, "Distributed combined authentication and intrusion detection with data fusion in high-security mobile ad hoc networks," *IEEE Trans. Veh. Tech.*, vol. 60, no. 3, pp. 1025 –1036, Mar. 2011.
[41] H. Y. Ong, K. Chavez, and A. Hong, "Distributed deep Q-learning," 2015. [Online]. Available: https://arxiv.org/abs/1508.04186
[42] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2011.
[43] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, vol. 703. Hoboken, NJ, USA: Wiley, 2007.
[44] M. Imani and U. M. Braga-Neto, "Control of gene regulatory networks with noisy measurements and uncertain inputs," *IEEE Trans. Control Netw. Syst.*, vol. 5, no. 2, pp. 760–769, Jun. 2018.
[45] A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf, "Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof," *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 38, no. 4, pp. 943–949, Aug. 2008.
[46] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjel, and G. Ostrovski, et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
[47] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[48] A. Martín, et al., "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," (2016). [Online]. Available: https://arxiv.org/abs/1603.04467
[49] Y. Zhao, W. Song, and Z. Han, "Social-aware data dissemination via device-to-device communications: Fusing social and mobile networks with incentive constraints," *IEEE Trans. Serv. Comput.*, vol. 12, no. 3, pp. 489–502, May–Jun. 2019.
[50] B. Kaufman and B. Aazhang, "Cellular networks with an overlaid device to device network," in *Proc. Asilomar Conf. Signals Syst. Comput.*, Oct. 2008, pp. 1537–1541.
[51] D. H. Lee, K. W. Choi, W. S. Jeon, and D. G. Jeong, "Two-stage semi-distributed resource management for device-to-device communication in cellular networks," *IEEE Trans. Wireless Commun.*, vol. 13, no. 4, pp. 1908–1920, Apr. 2014.
[52] P. Blasco and D. Gunduz, "Learning-based optimization of cache content in a small cell base station," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2014, pp. 1897–1903.
[53] J. Wu, S. Guo, J. Li, and D. Zeng, "Big data meet green challenges: big data toward green applications," *IEEE Syst. J.*, vol. 10, no. 3, pp. 888–900, Sep. 2016.
[54] J. Wu, S. Guo, J. Li, and D. Zeng, "Big data meet green challenges: Greening big data," *IEEE Syst. J.*, vol. 10, no. 3, pp. 873–887, Sep. 2016.



Ying He (S'16) received the BS degree from Dalian Ocean University, Dalian, China, and the MS Degree from Dalian University of Technology, Dalian, in 2011 and 2015, respectively, both in communication and information systems. She is currently working toward the PhD degree in the Department of Systems and Computer Engineering, Carleton Unviersity, Canada. Her current research interests include wireless systems, security, big data and machine learning. She is a student member of the IEEE.

Chengchao Liang (S'15-M'17) received the BEng and MEng in Communication and Information Systems from Chongqing University of Posts and Telecommunications, China, in 2010 and 2013, respectively, and the PhD in Electrical and Computer Engineering from Carleton University, Canada in 2017. He was awarded the Senate Medal. He is currently a Postdoctoral Fellow in the Department of Systems and Computer Engineering, Carleton University supported by Mitacs and Huawei Technologies Canada. He is serving and has served as the reviewer and the TPC member for many IEEE journals and conferences. His research interests include mobile networking, caching and computing with emphasis on cross-layer/system optimization, and convex theory. He is a member of the IEEE.

F. Richard Yu (S'00-M'04-SM'08-F'18) received the PhD degree in electrical engineering from the University of British Columbia (UBC) in 2003. From 2002 to 2006, he was with Ericsson (in Lund, Sweden) and a start-up in California, USA. He joined Carleton University in 2007, where he is currently a Professor. He received the IEEE Outstanding Service Award in 2016, IEEE Outstanding Leadership Award in 2013, Carleton Research Achievement Award in 2012, the Ontario Early Researcher Award (formerly Premiers Research Excellence Award) in 2011, the Excellent Contribution Award at IEEE/IFIP TrustCom 2010, the Leadership Opportunity Fund Award from Canada Foundation of Innovation in 2009 and the Best Paper Awards at IEEE ICNC 2018, VTC 2017 Spring, ICC 2014, Globecom 2012, IEEE/IFIP TrustCom 2009 and Int'l Conference on Networking 2005. His research interests include wireless cyber-physical systems, connected/autonomous vehicles, security, distributed ledger technology, and deep learning. He serves on the editorial boards of several journals, including Co-Editor-in-Chief for Ad Hoc & Sensor Wireless Networks, Lead Series Editor for IEEE Transactions on Vehicular Technology, IEEE Transactions on Green Communications and Networking, and IEEE Communications Surveys & Tutorials. He has served as the Technical Program Committee (TPC) Co-Chair of numerous conferences. He is a registered Professional Engineer in the province of Ontario, Canada, a Fellow of the Institution of Engineering and Technology (IET), and a Fellow of the IEEE. He is a Distinguished Lecturer, the Vice President (Membership), and an elected member of the Board of Governors (BoG) of the IEEE Vehicular Technology Society.



Zhu Han (S'01-M'04-SM'09-F'14) received the BS degree in electronic engineering from Tsinghua University, in 1997, and the MS and PhD degrees in electrical and computer engineering from the University of Maryland, College Park, in 1999 and 2003, respectively. From 2000 to 2002, he was an R&D Engineer of JDSU, Germantown, Maryland. From 2003 to 2006, he was a research associate with the University of Maryland. From 2006 to 2008, he was an assistant professor with Boise State University, Idaho. Currently, he is a professor with the Electrical and Computer Engineering Department as well as with the Computer Science Department, University of Houston, Texas. His research interests include wireless resource allocation and management, wireless communications and networking, game theory, big data analysis, security, and smart grid. He received an NSF Career Award in 2010, the Fred W. Ellersick Prize of the IEEE Communication Society, in 2011, the EURASIP Best Paper Award for the *Journal on Advances in Signal Processing* in 2015, IEEE Leonard G. Abraham Prize in the field of Communications Systems (best paper award in IEEE JSAC) in 2016, and several best paper awards in IEEE conferences. Currently, He is an IEEE Communications Society Distinguished Lecturer. He is a fellow of the IEEE.