Video Surveillance on Mobile Edge Networks—A Reinforcement-Learning-Based Approach

Haoji Hu, *Member, IEEE*, Hangguan Shan, *Member, IEEE*, Chuankun Wang, Tengxu Sun, Xiaojian Zhen, Kunpeng Yang, Lu Yu, *Member, IEEE*, Zhaoyang Zhang, *Member, IEEE*, and Tony Q. S. Quek, *Fellow, IEEE*

***Abstract*—Video surveillance systems or Internet of Multimedia Things are playing a more and more important role in our daily life. To obtain useful surveillance information timely and accurately, not only image recognition algorithms but also computing and communication resources can be bottlenecks of the whole system. In this article, taking face recognition application as an example, we study how to build video surveillance systems by utilizing mobile edge computing (MEC), one of the 5G's key technologies. Specifically, to achieve high recognition accuracy and low recognition time, we design image recognition algorithms for both the camera sensor and MEC server, and utilize the action-value methods to train actions of the system by jointly optimizing offloading decision and image compression parameters. The experimental results show the advantages of the proposed system for enabling communication environment-adaptive, efficient, and intelligent video surveillance.**

***Index Terms*—ϵ-greedy, action-value method, intelligent video surveillance systems, mobile edge computing (MEC).**

I. Introduction

IN RECENT years, a new trend emerges to remove computation from clouds to the network edges, which is in combination with the popularity of mobile devices and the exponential growth of mobile Internet. The area of mobile edge computing (MEC) aims at performing computation tasks at mobile edge servers, which usually connect directly with base stations (BSs) or access points (APs) [2]. Thanks to the short distance between end devices/user equipments and mobile edge servers, computation-intensive and latency-critical tasks are capable to be supported by MEC-based technologies [1]–[3].

In this article, we concentrate the MEC application on video analysis service, which has a broad range of applications, such as face recognition, vehicle recognition, and video surveillance, for which the basic operation includes object detection and classification. In these applications, tasks are first captured by front-end devices (i.e., camera sensors), then transmitted to back-end devices (i.e., MEC servers), if any, for further computation. For data storage and high confidence level image processing, a cloud data center can also be deployed. By assuming that there is mutual communication between back and front ends, the problem of MEC-based video analysis arises as obtaining useful surveillance information timely and accurately from a large amount of video data by cooperation between back-end and front-end devices [4]. As indicated by ETSI MEC ISG in [5], the bottleneck to solve this problem is not only from the lack of real-time and efficient image recognition algorithms but also from the limitations of computing and communication resources for video surveillance applications.

Existing MEC-based designs for video analysis tend to assign all analysis tasks either at the video capturing devices (the front end) or at the mobile edge servers (the back end). However, both strategies have their intrinsic problems. Assigning all analysis at the front end is unacceptable in reality, because for each camera sensor, we need to allocate enough computation resources to fulfill video analysis algorithms, which normally have a high computational complexity, making the cost of the system very high [4]. With the development of video analysis algorithms, especially recently more and more frequently adopted deep learning algorithms, which require massive computational complexity [6]–[8], MEC-based approaches tend to move the analysis tasks away from the front end to the back end [9], [10]. However, moving video analysis tasks to the back end may consume a great amount of network bandwidth to transmit video streams from camera sensors to servers, which is not realistic in situations when bandwidth resources are expensive. To alleviate the above

Manuscript received July 26, 2019; revised September 26, 2019, November 13, 2019, and December 27, 2019; accepted January 9, 2020. Date of publication January 23, 2020; date of current version June 12, 2020. This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB1801104; in part by the National Natural Science Foundation Program of China under Grant 61771427, Grant 61725104, and Grant U1709214; in part by the SUTD-ZJU Research Collaboration under Grant SUTD-ZJU/RES/05/2016; in part by the SUTD Growth Plan Grant for AI; in part by the Ng Teng Fong Charitable Foundation in the form of ZJU-SUTD IDEA Grant; in part by the SUTD-ZJU IDEA Grant for Visiting Professor under Grant 201804; and in part by the Huawei Technologies Company Ltd. under Grant YBN2018115223. *(Corresponding author: Hangguan Shan.)*

Haoji Hu and Hangguan Shan are with the College of Information Science and Electronic Engineering, Zhejiang Provincial Key Laboratory of Information Processing and Communication Networks, and SUTD-ZJU IDEA, Zhejiang University, Hangzhou 310027, China, and also with the Zhejiang Laboratory, Hangzhou 310000, China (e-mail: haoji_hu@zju.edu.cn; hshan@zju.edu.cn).

Chuankun Wang, Tengxu Sun, Xiaojian Zhen, Kunpeng Yang, Lu Yu, and Zhaoyang Zhang are with the College of Information Science and Electronic Engineering, Zhejiang Provincial Key Laboratory of Information Processing and Communication Networks, and SUTD-ZJU IDEA, Zhejiang University, Hangzhou 310027, China (e-mail: 21760623@zju.edu.cn; 21860235@zju.edu.cn; xiaojianzhen@zju.edu.cn; 21860162@zju.edu.cn; yul@zju.edu.cn; ning_ming@zju.edu.cn).

Tony Q. S. Quek is with the Information Systems Technology and Design Pillar, Singapore University of Technology and Design, Singapore (e-mail: tonyquek@sutd.edu.sg).

Digital Object Identifier 10.1109/JIOT.2020.2968941

problems, Anjum *et al.* [11] proposed MEC-based face recognition by assigning the face detection task to the front end and face recognition to the back end. Though the approach saves bandwidth resources greatly because only face images are transmitted instead of the whole video streams, bandwidth consumption can be further reduced if face images of high quality can be analyzed efficiently with appropriately designed recognition algorithms at the front end, thus avoiding being transmitted to the back end.

In this article, taking face recognition as an application example, we focus on designing an MEC-enabled smart camera system. Specifically, we consider a scenario in which both the camera sensor and the MEC server at the BS have computation resources. However, the former has much less computation resources than the latter. Thus, both of them can fulfill face recognition task with image processing algorithms tailored for them; yet, depending on the application requirements and communication environments, the camera sensor also can offload a face recognition task to the MEC server. We assume that recognition accuracy and recognition time overhead are two main application requirements, as they are important for many social security applications. We put the tradeoff between recognition accuracy and process time by making joint decisions on offloading and image compression setting into a coherent optimization scheme. The contributions of this article are summarized as follows.

First, to design an MEC-enabled smart camera system, in which both the camera sensor and MEC server have recognition abilities, we design different face recognition algorithms for the camera sensor and MEC server, i.e., local binary patterns (LBPs)-based algorithm and convolutional neural networks (CNNs)-based algorithm, respectively. The reason to design independent algorithms for them rather than using the traditional computation offloading strategy (parallel computing with data interaction) is that CNN in those powerful image recognition algorithms is a cascaded layer-by-layer structure and the data amount is huge for each layer. Thus, there will be huge data interaction between the camera sensor and MEC server (thus the BS) if following the traditional strategy. For our approach, recognition results are first generated by the front end, and only when it is not sure about the results, it will send face images to the back end. Our approach is especially valuable when bandwidth resources between front and back ends are limited.

Second, to balance recognition accuracy and bandwidth consumptions, we combine recognition algorithms, image compression, and data transmission into an integrated framework using the action-value method, a reinforcement-learning-based approach.[1] Specifically, the state space of the proposed decision system is designed to be composed of the confidence level (a parameter output by the recognition algorithm at the front end to represent how confident about the recognition result) and the channel quality (measured by the front end). Furthermore, aiming at improving recognition accuracy while reducing recognition time overhead, we propose a class of new reward functions related to both recognition result and time overhead.

Finally, we design control algorithms for both stationary and nonstationary channel conditions. For stationary channel conditions, the action-value method with softmax probability assignments is utilized to maximize the reward function. For nonstationary channel conditions, the action-value algorithm with ϵ-greedy probability assignments is used because the ϵ-greedy algorithm can make the state–action table converge faster in nonstationary channel conditions, with the compromise of lower cumulative rewards. Training results based on real face image data set not only show the convergence of the proposed action-value optimizer but also shed some light on the relationship between the best action and the changing system states. Furthermore, we do extensive simulations to compare the performance of the proposed scheme with other decision-making policies and validate its advantage in improving the cumulative reward.

The remainder of this article is organized as follows. We discuss the related works in Section II. In Section III, we present our MEC-enabled camera system and the algorithm design for the whole system, including image processing at both the camera sensor and the MEC server, and the action-value optimizer. The experimental results are given in Section IV, followed by the conclusions in Section V.

II. Related Works

Video surveillance is a common technology used in public affairs, focusing its use mostly on public transportation, crime detection, and security intervention [13]. Surveillance applications require significant computing and storage resources to handle massive contextual data created by video sensors. According to the estimation of Cisco Corporation, the global Internet video surveillance traffic will increase sevenfold between 2017 and 2022, accounting for 3% of all Internet video traffic by 2022 and up from 2% in 2017 [14]. Thus, it is important to handle the massive data produced by video cameras in new ways. The MEC paradigm provides excellent flexibility and is also scalable corresponding to the increasing number of surveillance cameras [15].

The main problem of recent MEC-based video surveillance approaches lies in two parts. First, the edge environment necessitates lightweight but robust video processing algorithms. However, in the area of face recognition, it is difficult to satisfy both of these contradictory requirements. Traditional face recognition methods include principle component analysis (PCA) [16], elastic graph matching (EGM) [17], LBP [18], etc. These are lightweight methods and easy to be deployed on resource-limited devices, but their accuracies are unacceptable. Face recognition has been revitalized in the era of deep learning. Much effort is devoted to design deep neural network architectures to obtain state-of-the-art accuracy on the famous labeled faces in the wild (LFW) benchmark [19], approaching human performance on the unconstrained condition for

[1]Although reinforcement learning was used in other MEC-based applications, e.g., Xiao *et al.* [12] designed a malware detection scheme with Q-learning for a mobile device to derive the optimal offloading rate without knowing the trace generation and the radio bandwidth model of other mobile devices, it is rarely implemented to a recognition system to balance the recognition accuracy and time complexity of the whole system.

the first time [20], [21]. Recent trend of face recognition research focuses on designing loss functions to further enhance the capability of neural networks. For example, along with FaceNet proposed by Google [22], triplet loss is introduced into face recognition and effectively improves recognition accuracy [22]. In recent years, other loss functions, such as center loss [7] and angular loss [23], are also proposed to further improve the accuracy of face recognition systems. Deep learning-based methods have dramatically improved the accuracy of the surveillance system, but require much more computation compared with the traditional methods. Recently, the community has recognized the importance of efficient models for mobile and embedded applications. Lightweight CNN architectures, such as MobileFaceNet [24], are implemented to achieve real-time performance on a mobile phone, but its computation is still five to ten times more than traditional methods.

The second problem of MEC-based video surveillance approaches is how to design and allocate communication paradigm among different levels of edge devices. There are many works proposing theoretical frameworks to design offloading policy for MEC (e.g., [25]–[27]). Zhang *et al.* [25] proposed an energy-optimal offloading policy, which aims at conserving energy for resource-constrained mobile devices. You *et al.* [26] studied resource allocation for a multiuser mobile edge computation offloading system based on time-division multiple access (TDMA) and orthogonal frequency-division multiple access (OFDMA) for minimizing the weighted sum mobile energy consumption. As it is NP-hard to find a centralized optimal solution to the multiuser computation offloading problem for MEC in a multichannel wireless interference environment, Chen *et al.* [27] studied distributed computation offloading decision-making problem from a game perspective. Recently, the idea of MEC has also been explored to build efficient field monitoring systems (e.g., [28] and [29]). In [28], the edge controller node in an MEC-based energy-efficient field monitoring system can send feedback messages to sensors to optimize sensing frequency, capture rate, and video quality, thus reducing the energy consumption of sensors and the traffic poured into the network. Leveraging a sensor-edge-cloud multitier architecture, where is the bear (WTB), an end-to-end, distributed data acquisition and analysis system for wildlife monitoring is proposed in [29]. However, for MEC-based face recognition systems, to the best of our knowledge, there exist no practical schemes, which are designed targeted for optimizing application requirements, such as recognition accuracy and process time together, by jointly considering the recognition algorithm design, and offloading and image compression setting.

III. System and Algorithm Design

The flowchart of the proposed system model is shown in Fig. 1. The system can be divided into three parts: 1) the front-end (camera) part; 2) the decision-making module; and 3) the back-end (server) part.

1) In the front-end part, face images are first taken by the camera, and then a simple recognition algorithm with low computational complexity is used to obtain a primary recognition result. In addition, the front end also generates a measure, which is called confidence level, to represent how confident the algorithm is about its recognition result.
2) The decision-making module takes the confidence level as input and meanwhile considers the communication channel conditions to decide whether to transmit the image to the back end and, if so, what the compression ratio is.
3) If the face image is not sent to the back end, the final recognition result is from the front end; otherwise, it is from the back-end MEC server. After obtaining the compressed face image from the front end via the wireless channel, the back end decompresses it and generates recognition result by a recognition algorithm which is more accurate but more complex than the front-end one.

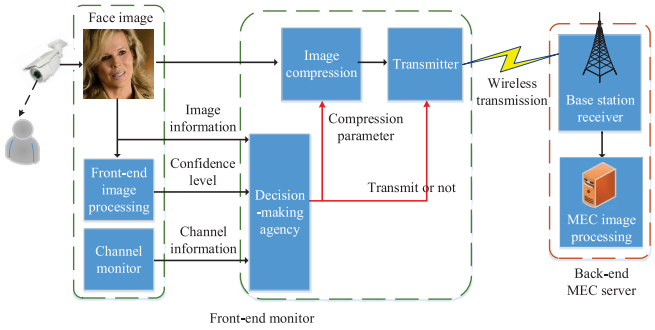

Fig. 1. System model of MEC-based face recognition.

It is obvious that if the recognition result of the front end is credible, e.g., with a high confidence level, it is not necessary to send images to the back-end MEC server, thus bandwidth resources between back and front ends can be saved. In the following, we elaborate the MEC-enabled camera system by introducing the back-end and front-end face recognition algorithms, the image compression algorithm, the calculation of the confidence level, and the decision-making module, respectively.

A. Front-End and Back-End Face Recognition Algorithms

The front-end face recognition algorithm adopted in this article is based on LBP [18], [30], which assigns a label to every pixel of an image by thresholding the 3×3 neighborhood of each pixel with the center pixel value and considering the result as a binary number. Then, the histogram of the labels can be used as a texture descriptor. For constructing the classifier, we divide the face image into 7×7 blocks, and use the uniform LBP to extract features in the 49 regions. For each block, the uniform LBP obtains a 59-D feature. The features of the 49 blocks are concatenated to finally form a 2891-D feature for each image. Recognition results are obtained by comparing the 2891-D features of two images using a weighted chi-square distance metric [18], where the LBP features of the 49 blocks are compared by weighted sum to indicate their relative importance for recognition. Specifically, supposing that the normalized LBP histograms of two images are $\mathbf{x}$ and $\mathbf{y}$,

TABLE I
COMPARISON OF THE LBP AND CNN FACE RECOGNITION ALGORITHMS

	Time on CPU (ms)	Recognition rate on LFW	TPR with FPR=0.1	TPR with FPR=1
LBP	5.85	65.72%	11.57%	22.15%
CNN	36.92	98.30%	96.50%	99.07%

then the weighted chi-square distance can be defined as

$$D(\mathbf{x}, \mathbf{y}) = \sum_{j,i} \omega_j \frac{(x_{i,j} - y_{i,j})}{x_{i,j} + y_{i,j}} \tag{1}$$

where indices i and j are referred to as the ith element of LBP corresponding to the jth local region (49 local regions in total), and ω_j is the weight for region j. Finally, the recognition results are obtained based on the distance calculated by (1). If the distance is less than a threshold, we classify them as the same person, otherwise, we classify them as different persons. The distance threshold is carefully chosen to obtain the highest recognition rate on training image pairs.

The baseline architecture of our back-end face recognition algorithm is ResNet28 proposed by [31]. Its CAFFE implementation is from GitHub.[2] We intensively decrease the number of convolutional kernels to 1/8 of the original network to save computation. The network also uses the center loss [7] combined with softmax loss as a supervision signal to train the deep model, which enhances the discriminative power of deeply learned features for recognition. We first train the network on the CASIA-Webface database, which is a large-scale data set, including about 10 000 subjects and 500 000 face images. After the training process converges, we further fine-tune the network by using our own IDCard face database (around 140 000 IDCard images). After training the CNN, recognition results are obtained by first extracting features from the network, and then comparing the Euclidean distance between images. If the distance is below a threshold, then we classify them as the same person, otherwise we classify them as different persons.

Table I compares the front-end LBP and back-end CNN algorithms. On the Intel Xeon CPU E5-2620 v4@2.10-GHz, LBP uses 5.85 ms to process one image pair while CNN uses 36.92 ms, which is around six times slower. The recognition rates on the LFW database [19], which is a popular database to test face recognition algorithms, are 65.72% and 98.30% for LBP and CNN, respectively. On our IDCard data set, we compare the true positive rate (TPR) of LBP and CNN when the false positive rate (FPR) equals 0.1% and 1% (basic introduction for TPR and FPR can be found in [32]). From these results, it is seen that CNN achieves a much better recognition rate than LBP.

B. Image Compression Algorithm

The image compression algorithm implemented in this article is X264. It is a free software library and application for encoding video streams into the H.264/MPEG-4 AVC compression format, and is released under the terms of the GNU GPL.[3] We need rate control schemes to decide how many bits will be used for each image. The rate control mode selected in this article is a constant rate factor (CRF), which allows the encoder to achieve a certain output quality for the whole file when output file size is of less importance. This provides a maximum compression efficiency with a single pass. By adjusting the quantizer for each frame, it obtains the bit rate needed to keep the requested quality level.

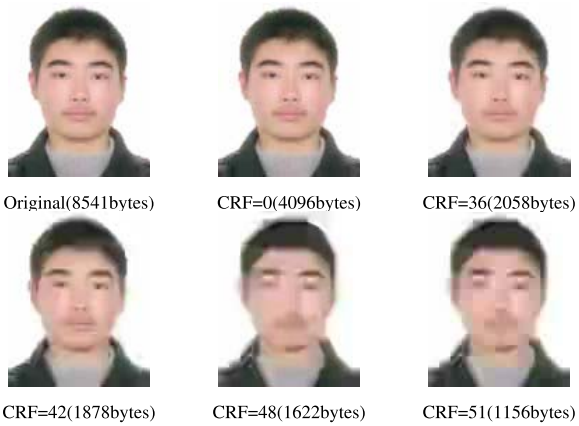

Fig. 2. Original and compressed images in different compression settings.

The range of the CRF scale is from 0 to 51, and a lower value generally leads to higher quality, where 0 refers to lossless, 23 is the default, and 51 is the worst quality possible. The range is exponential, so increasing the CRF value +6 results in roughly half the bit rate decrease. However, because CRF is to control the output quality level, it cannot directly obtain a specific bit rate after compression. Fig. 2 shows one face image and its five compressions when CRF equals 0, 36, 42, 48, and 51, respectively. It can be seen that the image gets blurred and file size gets smaller when CRF increases.

C. Calculation of the Confidence Level

As indicated in the system model, the agency uses the confidence level denoted by η to decide whether the face image obtained from the front end needs to be sent to the back end. The calculation of the confidence level is based on the probability estimation of image pair distances.

Fig. 3 illustrates the process of calculating the confidence level. Supposing that there are N training image pairs. We use d_i $(i = 1, 2, \ldots, N)$ to represent the distances obtained by (1). As shown in Section III-A, a threshold (the yellow line in Fig. 3) is selected to ensure the best recognition rate on training pairs. All distances are divided into M bins based on their values. The bins are unequally assigned to ensure that there are approximately the same number (N/M) of distances in each bin. The distances are represented as red dots (labeled as the same person) and green dots (labeled as different persons) in Fig. 3. The confidence level in each bin is estimated as the recognition rate of all distances in that bin, i.e., for the bin less than the threshold, the confidence level is obtained with the number of red dots divided by the number of all dots

[2]https://github.com/ydwen/caffe-face/tree/caffe-face/face_example

[3]http://git.videolan.org/git/x264.git

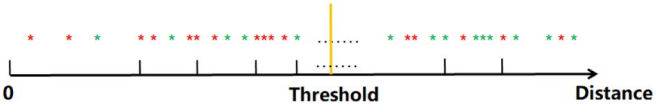

Fig. 3. Calculation of the confidence level of the front-end algorithm.

in that bin; and for the bin greater than the threshold, confidence level is obtained with the number of green dots divided by the number of all dots in that bin.

D. Action-Value-Based Decision-Making Module

The optimization goal of the decision-making module is to maximize the recognition accuracy while reducing the recognition time overhead under fixed computation and communication resources. To achieve this goal, the decision-making module at the front end first needs to determine whether to transmit the image to the back end. If it decides to transmit, it also needs to choose a specific compression rate (i.e., choose a CRF) to compress the image during transmission. All decisions are made based on states of the confidence level and detected channel quality.

We use reinforcement learning algorithms for automatic decision making based on training samples. In this article, the action-value method, as one of the reinforcement learning algorithms, is utilized for decision making [33]. The action-value method selects one action from a finite set of actions based on the current state of the system with the goal of maximizing a predefined reward function. The algorithm eventually finds an optimum given some training data, in the sense that the expected value of the cumulative average reward returned over all successive steps, starting from the current state, is the maximum achievable.

For our decision-making module, we define the state space, action space, and reward functions as follows.

State Space: For each image pair, we have 2-D state space $S = (\eta, h(t))$. The first dimension is the confidence level, which is calculated at the front end using the method in Section III-C. The second dimension $h(t)$ represents the wireless channel gain, e.g., the greater value of $h(t)$, the better the channel so that it requires less time to transmit a fixed amount of data. In this article, we assume $h(t) = d^{-\alpha} g_s(t)$, where d is the distance between the camera sensor and BS, α is the path loss exponent, and $g_s(t)$ is the small-scale fading satisfying Rayleigh fading with unit mean. However, as the face image could be transmitted over multiple durations of coherent time, the decision-making module only has the channel gain of the current time t before transmission. To limit the searching space, the state spaces are discretized so that η and $h(t)$ take discrete values.

Action Space: The action space is also a 2-D vector which is represented by A (whether_transmit, CRF). The first dimension whether_transmit is a binary to indicate whether the front end needs to transmit the image pair to the back end, where 1 represents "transmit" and 0 represents "not transmit." The second dimension is only meaningful when whether_transmit = 1. It represents the rate control factor CRF value defined in Section III-B to compress the image pair before transmission. Similar to the state space, the CRF is also discretized to take a limited amount of values. Supposing that we have K actions in total, we use A_i $(i = 1, 2, \ldots, K)$ to represent these actions.

Reward Function: We propose a class of reward functions of the system as

$$R(Y, T) = \frac{Y}{\phi(\beta, T)} \tag{2}$$

where Y is a binary bit to indicate whether the whole system recognizes a specific image pair correctly, T is the process time of the image pair in the system, and $\phi(\beta, T)$ is a weighting function used to balance the weight of time on image processing (computation) and image transmission (communication) by tuning a positive parameter β. We specify each of them as follows.

For indicator Y, it equals 1 for correct recognition and 0 for incorrect recognition. More specifically, if the front end decides not to transmit the image pair to the back end (whether_transmit = 0), Y is decided by the front-end recognition algorithm; otherwise, the image pair is transmitted to the back end and Y is decided by the back-end algorithm.

For process time T, it consists of three parts

$$T = T_1 + T_2 + T_3. \tag{3}$$

Here, T_1 is the process time of the front-end LBP algorithm. T_2 refers to the image transmission time over the wireless channel, which is a function of the file size and the channel condition $h(t)$. Actually, T_3 can be further divided into three parts. The first part is the process time of the compression algorithm at the front end; the second part is the process time of extracting images at the back end; and the third part is the process time of the back-end CNN recognition algorithm. For simplicity, we assume that T_3 is a fixed value for all image pairs. We should note that when the front end decides not to transmit images to the back end (whether_transmit = 0), we need to set both T_2 and T_3 to zero.

For weighting function $\phi(\beta, T)$, in the experiment we set up three different forms, namely, exponential form ϕ_{exp}, linear form ϕ_{lin}, and logarithmic form ϕ_{log}, which are given as follows:

$$\phi_{\text{exp}}(\beta_{\text{exp}}, T) = e^{\beta_{\text{exp}} T} \tag{4}$$

$$\phi_{\text{lin}}(\beta_{\text{lin}}, T) = \beta_{\text{lin}} T + 1 \tag{5}$$

$$\phi_{\text{log}}(\beta_{\text{log}}, T) = (\beta_{\text{log}} T + 1) + 1. \tag{6}$$

Accordingly, we have three forms of reward function, namely, exponential form reward function R_{exp}, linear form reward function R_{lin}, and logarithmic form reward function R_{log}, respectively.

In (2), we set the form of reward function $R(Y, T)$ as the recognition result Y divided by $\phi(\beta, T)$ because our first priority is correct recognition, and only when the recognition result is correct, we require the process time as small as possible. Since $\phi(\beta, T)$ in (4)–(6) is set as a monotonically increasing function, the reward is greater when T is smaller, which is exactly our goal for the reward function. In addition,

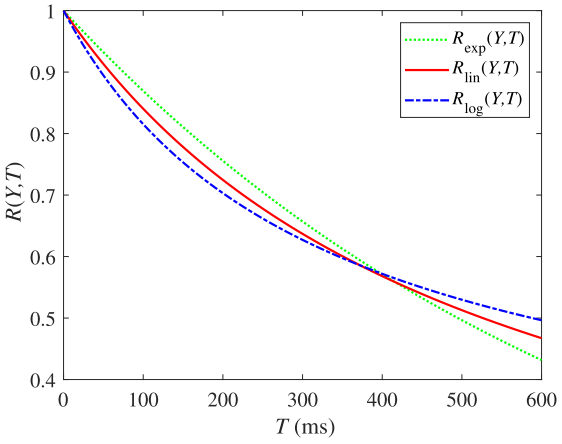

Fig. 4. Three different forms of reward functions.

the three forms of reward function punish the process time T to different degrees. We will compare their effects to the system in Section IV. Our purpose to set three different reward functions is to show that the proposed system is relatively robust and stable to different types of rewards. Fig. 4 shows the three forms of reward function by setting $\beta_{\text{exp}} = 0.0014$, $\beta_{\text{lin}} = 0.0019$, and $\beta_{\text{log}} = 0.0017$, respectively. The parameters are deliberately tuned so that the curves are in the same range for comparison.

1) Action-Value Method for Stationary Channel Conditions: We first consider the scenario that the wireless channel is stationary. Specifically, we consider the average channel gain between the camera sensor and BS $E[h(t)]$ is unchanged. The action-value algorithm needs to calculate a state–action table represented as $Q(S, A)$, which means the cumulative average reward at the current time when taking action A at state S. With simple deductions [33], we can obtain the updating equation of $Q(S, A)$ as

$$Q^*(S, A) = Q(S, A) + \frac{1}{n(S, A) + 1}[R(Y, T) - Q(S, A)] \quad (7)$$

where $[1/(n(S, A) + 1)]$ is a positive step-size parameter that represents the learning rate, with $n(S, A)$ denoting the number of action A being selected at state S in the previous progress. The expression in the square brackets can be seen as an error in the estimate which indicates the direction of the estimate convergence. This method finally makes $Q(S, A)$ converge to the cumulative average rewards when taking action A at state S. The updating method of (7) is similar with the n-armed bandit problem in reinforcement learning [33], provided that in the n-armed bandit problem, the rewards are interactively decided by the action and the environment; while in our method, the rewards are decided by the current state, current action, and the environment. In addition, the original n-armed bandit problem typically generates a specific reward for each action, but our method generates the reward for each action from a distribution which is associated with the channel conditions.

While training the decision-making module, two methods of selecting actions can be used. If the training set is small or in the early stage of training, we can directly traverse all actions in state S; otherwise, if the training set is large or there is a clear distinction in Q-value between different actions, we can apply softmax to improve the training efficiency [34] in which action A_i is Monte Carlo sampled from probability

$$p(A_i) = \frac{\exp(Q(S, A_i))}{\sum_{j=1}^{K} \exp(Q(S, A_j))}. \quad (8)$$

By the above definitions, the whole algorithm is summarized in Algorithm 1. Here, we apply softmax to train the state–action table. After the state–action table is obtained, for a testing image pair, we first calculate its state $S = (\eta, h(t))$. Then, we can select the action which maximizes the Q-value under that state

$$A = \underset{i=1,2,\ldots,K}{\arg\max}\ Q(S, A_i). \quad (9)$$

2) ϵ-Greedy Algorithm for Nonstationary Channel Conditions: For stationary channel conditions, the action which Algorithm 1 outputs for a specific state should maximize the cumulative reward for the state and always keep unchanged. However, for nonstationary channel conditions, e.g., the scenario that the average channel gain between the camera sensor and the BS can change with time due to, for example, the moving camera sensor, we need to design a new algorithm which can track the change.

To this end, we implement the ϵ-greedy algorithm to make the system adapted to the changes of communications environments. In dealing with nonstationary environment, the *dilemma of exploration and exploitation* [33] becomes prominent. If the agent always keeps on exploiting optimized actions in the current stage, the obtained reward may not be optimum because the changing environment would also change the optimized actions. However, if the agent always tries to find actions which maximize the cumulative reward function, there would be a great amount of reward loss during such an exploration process. Thus, the ϵ-greedy algorithm is proposed to balance the ratio of exploitation and exploration.

Specifically, in our application, the decision-making module is first initialized with a trained state–action table which is obtained by Algorithm 1. Then the agent selects an action

Algorithm 1 Action-Value Algorithm for Stationary Channel Conditions

```
Input the training set D consisting of several image pairs and their labels;
Initialize Q(S, A_i) = 0 for all states S;
Initialize n(S, A_i) = 0 for all states S;
repeat
    for each image pair in training set D do
        Obtain its state S = (η, h(t));
        Calculate p(A_i) for i = 1, 2, ..., K by (8);
        Select A by Monte Carlo sampling from p(A_i);
        Based on action A, obtain Y and T from reality;
        n(S, A) = n(S, A) + 1;
        Update Q(S, A) by (7);
    end for
until Q(S, A) converges for all S and A;
Output the state-action table Q(S, A).
```

Algorithm 2 ϵ-Greedy Algorithm for Nonstationary Channel Conditions

Input the training set D consisting of several image pairs and their labels;
Input the probability hyper-parameter ϵ;
Initialize the trained state-action table using Algorithm 1 for all states $Q(S, A)$;
Initialize $n(S, A_i) = 0$ for all states S, $i = 1, 2, \ldots, K$;
repeat
  for each image pair in training set D **do**
    Obtain its state $S = (\eta, h(t))$;
    Generate a uniformly distributed random value $c \in (0, 1)$;
    if $c < \epsilon$ **then**
      Select an action A randomly from $\{A_1, A_2, \ldots, A_K\}$ with equal probability;
    else
      Select the action A greedily according to (9);
    end if
    Based on action A, obtain Y and T from reality;
    $n(S, A) = n(S, A) + 1$;
    Update $Q(S, A)$ by (7);
  end for
until $Q(S, A)$ converges for all S and A;
Output the state-action table $Q(S, A)$.

randomly with a fixed probability ϵ (the exploration process), and selects the action greedily which offers the maximum of the current state–action table with probability $1 - \epsilon$ (the exploitation process). Finally, to capture the change of the situation, the state–action table is also updated. The reason why the ϵ-greedy algorithm can catch up with the changes of environments is that actions are randomly taken during the exploration process, which gives the opportunity that the state–action table can be updated according to the modification of the environment. The algorithm is summarized in Algorithm 2.

IV. Experimental Results

A. Face Database and Parameter Settings

The training face data set in our experiment is composed of 3200 positive pairs and 3200 negative pairs. Each positive pair contains an ID card image and a lifestyle image of the same person, while each negative pair contains an ID card image and a lifestyle image of two different persons. Some samples from the training set are shown in Fig. 5.

Here, for face detection and alignment, we use multitask cascaded CNNs (MTCNN) [35] to automatically crop the face region of each image to extract the face feature. When testing the performance of the trained system, we use 1600 testing pairs which are different from the training set, including both positive and negative pairs.

If running on the same device, the face recognition algorithm used for the front end is much faster than that for the back end (see Table I). However, because the performance of CPU in the front end is usually much lower than the CPU in the back end, in the simulation, we assume both the front end and back end need 40 ms to finish face recognition for one image, i.e., $T_1 = T_3 = 40$ ms. For the image transmission time over the wireless link T_2, we model it according to Shannon's theory

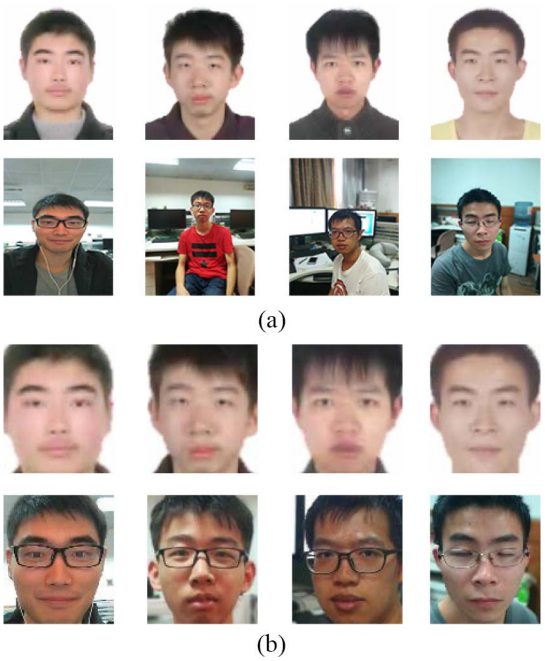

Fig. 5. Both in (a) and (b), the images in the first row is from ID card pictures, and the second row is from corresponding lifestyle pictures.

$$T_2 = \arg \min_{M \le \int_0^{\xi} B \log_2\left(1+\frac{P \cdot h(t)}{N_0}\right)} \xi \tag{10}$$

where ξ is a dummy variable representing the image transmission time, and M, B, P, and N_0 denote the size of compressed face image, the transmission bandwidth, the camera sensor's transmit power, and the power of the additive white Gaussian noise (AWGN), respectively. In the simulation, B, P, and N_0 are set as 100 KHz, 15 dBm, and -97 dB, respectively. In terms of the wireless channel, for the stationary channel conditions, we set $d = 30$ m and $\alpha = 3.5$ in our simulation. For the nonstationary channel conditions, we consider that the average channel gain $E[h(t)]$ can vary due to the camera sensor's movement. Besides, for both conditions, we assume that the small-scale fading remains unchanged in a time interval equal to 10 ms, but after the time interval, it can change according to Jakes' fading model [36]. When applying Jakes' fading model to update small-scale fading in stationary (nonstationary) channel conditions, we set the channel coherence time equal to 30.5 ms (inversely proportional to Doppler spread). Furthermore, without any optimization on channel state division, we divide $h(t)$ randomly into five intervals, denoted as $h_1 = d^{-\alpha} \times [0, 0.53)$, $h_2 = d^{-\alpha} \times [0.53, 0.81)$, $h_3 = d^{-\alpha} \times [0.81, 1.08)$, $h_4 = d^{-\alpha} \times [1.08, 1.43)$, and $h_4 = d^{-\alpha} \times [1.43, \infty)$. Similarly, we also divide the confidence level η into five intervals: $[0, 0.6)$, $[0.6, 0.7)$, $[0.7, 0.8)$, $[0.8, 0.9)$, and $[0.9, 1]$, denoted, respectively, by η_i, $i = 1, 2, \ldots, 5$. We choose 13 CRF values as our CRF space: 0, 18, 24, 30, 36, 38, 40, 42, 44, 46, 48, 50, and 51.

TABLE II
RELATIONSHIP BETWEEN RECOGNITION ACCURACY AND CONFIDENCE LEVEL FOR THE LBP-BASED ALGORITHM AT THE FRONT END

CL	η_1	η_2	η_3	η_4	η_5
Accuracy	53.0%	62.0%	73.0%	84.0%	93.0%

TABLE III
RELATIONSHIP BETWEEN RECOGNITION ACCURACY AND COMPRESSION SETTING FOR THE CNN-BASED ALGORITHM AT THE BACK-END MEC

Compression setting	Original	CRF =0	CRF =36	CRF =48	CRF =51
Accuracy	99.5%	98.9%	98.4%	87.7%	80.0%

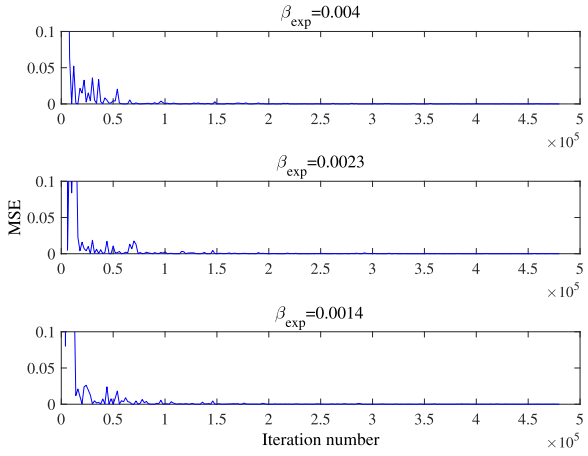

Fig. 6. Convergence of the MSE of the state–action table by using the exponential form reward function.

TABLE IV
STATE–ACTION TABLE UNDER THE EXPONENTIAL FORM REWARD FUNCTION WHEN $\beta_{exp} = 0.001$

A_i \ $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 36)	(1, 36)	(1, 36)	(1, 36)	(1, 30)
η_2	(1, 36)	(1, 36)	(1, 36)	(1, 36)	(1, 36)
η_3	(1, 38)	(1, 38)	(1, 38)	(1, 38)	(1, 38)
η_4	(1, 42)	(1, 42)	(1, 38)	(1, 38)	(1, 36)
η_5	(0, /)	(0, /)	(0, /)	(0, /)	(1, 44)

TABLE V
STATE–ACTION TABLE UNDER THE EXPONENTIAL FORM REWARD FUNCTION WHEN $\beta_{exp} = 0.0014$

A_i \ $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 36)	(1, 36)	(1, 36)	(1, 36)	(1, 36)
η_2	(1, 38)	(1, 38)	(1, 38)	(1, 36)	(1, 36)
η_3	(1, 38)	(1, 38)	(1, 38)	(1, 38)	(1, 38)
η_4	(0, /)	(0, /)	(1, 42)	(1, 40)	(1, 40)
η_5	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)

TABLE VI
STATE–ACTION TABLE UNDER THE EXPONENTIAL FORM REWARD FUNCTION WHEN $\beta_{exp} = 0.002$

A_i \ $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 38)	(1, 38)	(1, 38)	(1, 38)	(1, 36)
η_2	(1, 38)	(1, 38)	(1, 38)	(1, 38)	(1, 38)
η_3	(1, 40)	(1, 40)	(1, 40)	(1, 40)	(1, 40)
η_4	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)
η_5	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)

The relationship between the front-end recognition accuracy and confidence level is tested and shown in Table II. We also choose four CRF values—0, 36, 48, and 51, and test their corresponding recognition accuracy at the back end. The results are shown in Table III.

B. Experiments of the Action-Value Algorithm

1) Exponential Form Reward Function: First, we show the convergence of the exponential form reward function during the training process. Fig. 6 shows the mean-square error (MSE) of the state–action table in adjacent epoches. It can be seen that generally, the Q-value is convergent when the iteration number is greater than 1.5×10^5. Moreover, the choices of β_{exp} do not influence the convergence of the Q-value, indicating the stability of the proposed action-value algorithm.

Tables IV–VI show the results of the decision-making module with three different β_{exp} values—0.001, 0.0014, and 0.002, respectively. In all three situations, the expected value $E[h(t)]$ of channel gains is fixed. It is noteworthy that the three different β_{exp} values represent three kinds of decisive inclination. When β_{exp} becomes smaller, the denominator $e^{\beta_{exp}T}$ of the reward function maps T to a smaller value, thus decreasing the influence of transmission time overhead and finally, resulting in the decision-making module taking more consideration about recognition accuracy. It can be observed that as β_{exp} becomes smaller, the decision-making module tends to choose a smaller CRF thus, a lower compression ratio to improve recognition accuracy at the cost of transmitting more data. Furthermore, when the state parameter $h(t)$ increases, which means a better channel condition is taken at a specific time t, the decision-making module tends to transmit the image more frequently and with a smaller CRF to improve recognition accuracy at the cost of bandwidth consumptions. It is noteworthy that the impact of channel state at decision time may reduce, if the image transmission time increases to become much larger than the channel coherence time, due to too frequent channel variations during the image transmission time. That is, to build a channel-aware MEC-based video surveillance system, an appropriate setting of channel bandwidth for image transmission with respect to both image size and channel coherence time is important. It is also observed from Tables IV–VI that with the increase of the confidence level η given by the front end, the decision-making module is more likely to choose not to transmit the image to the back end (i.e., not to use the MEC server) because of the time overhead of image transmission. However, it is noteworthy that when the decision-making module chooses to transmit the image to the MEC server, the action selected is not determined by the confidence level η, which means the correlation between η and selected compression parameter CRF is weak when the image is transmitted.

We also do more experiments on the testing data set and compare the performance the action-value policy (AVP) with

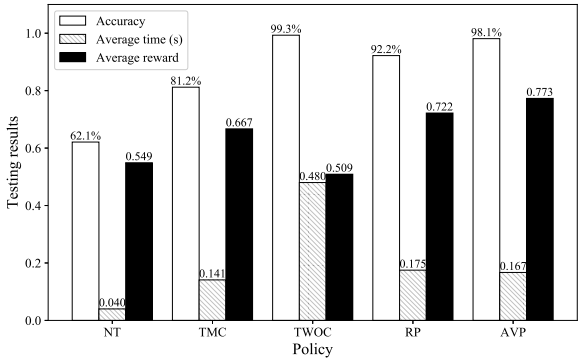

Fig. 7. Testing results with different policies under the exponential form reward function when $\beta_{\text{exp}} = 0.0014$.

the following four policies—no transmission (NT), transmit with maximum compression (TMC), i.e., CRF = 51, transmit without compression (TWOC), and random policy (RP). Fig. 7 shows the testing results in terms of recognition accuracy, average time, and average reward from all tested 1600 image pairs. It can be observed that the average reward obtained by AVP is the greatest, which is not surprising because the AVP specially aims at maximizing this reward. The accuracy of AVP is a little bit lower than that of TWOC. This is because TWOC always chooses to transmit the original images to maximize recognition accuracy. However, TWOC, of course, introduces much more time overhead than AVP. As compared with AVP, NT, and TMC consume much less time, because they either do not use the MEC server or only transmit images with the lowest quality to the MEC server, so their accuracies are lower than AVP.[4] It is also noteworthy that the performance of AVP and RP are very close. The recognition accuracy (time-complexity) of AVP is slightly better than RP in 5.9% (8.0 ms). The reason is that the designed CNN algorithm for the MEC server performs very well. For different compression settings, as shown in Table III, it always generates a relatively high value of recognition accuracy, which means that the back-end recognition algorithm plays a key role in the entire system.

2) Linear and Logarithmic Form Reward Functions: We repeat the above experiments by using the linear form reward function and the logarithmic form reward function. For the linear form reward function, we study three different β_{lin} values—0.0017, 0.0019, and 0.0023; and for the logarithmic form reward function, we also compare three different β_{log} values—0.001, 0.0015, and 0.0017. Tables VII–IX show the results of decision-making module using the linear form reward function, and the testing results compared with other polices are shown in Fig. 8. Similarly, Tables X–XII show the results of decision-making module using the logarithmic form reward function, and the corresponding testing results are given in Fig. 9.

From the above experiments, we can summarize the following conclusions. First, for all the three reward functions, when β becomes smaller, the decision-making module tends to choose a smaller CRF to compress the image with a lower compression ratio to improve recognition accuracy at the cost of transmitting more data. Second, when the state parameter $h(t)$ increases which indicating a better channel condition, the decision-making module tends to transmit images more frequently to the back end and with a smaller CRF to improve the recognition accuracy. Third, with the increase of the confidence level given by the front end, the decision-making module is more likely to choose not to transmit the image to the back end because of the time overhead of image transmission.

[4]It is noted from the aforementioned comparisons that due to the reward function design AVP provides a way to achieve a better balance between the recognition accuracy and process time. However, if latency-critical applications which have strict delay requirements should be supported, new algorithms integrated with delay constraints should be devised.

TABLE VII
STATE–ACTION TABLE UNDER THE LINEAR FORM REWARD FUNCTION WHEN $\beta_{\text{lin}} = 0.0017$

A_i \ $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 36)	(1, 36)	(1, 36)	(1, 36)	(1, 36)
η_2	(1, 38)	(1, 38)	(1, 36)	(1, 36)	(1, 36)
η_3	(1, 38)	(1, 38)	(1, 38)	(1, 38)	(1, 36)
η_4	(1, 42)	(1, 42)	(1, 42)	(1, 42)	(1, 36)
η_5	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)

TABLE VIII
STATE–ACTION TABLE UNDER THE LINEAR FORM REWARD FUNCTION WHEN $\beta_{\text{lin}} = 0.0019$

A_i \ $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 36)	(1, 36)	(1, 36)	(1, 36)	(1, 36)
η_2	(1, 38)	(1, 38)	(1, 36)	(1, 36)	(1, 36)
η_3	(1, 38)	(1, 38)	(1, 38)	(1, 38)	(1, 38)
η_4	(0, /)	(1, 42)	(1, 42)	(1, 42)	(1, 42)
η_5	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)

TABLE IX
STATE–ACTION TABLE UNDER THE LINEAR FORM REWARD FUNCTION WHEN $\beta_{\text{lin}} = 0.0023$

A_i \ $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 36)	(1, 36)	(1, 36)	(1, 36)	(1, 36)
η_2	(1, 38)	(1, 38)	(1, 38)	(1, 36)	(1, 36)
η_3	(1, 38)	(1, 38)	(1, 38)	(1, 38)	(1, 38)
η_4	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)
η_5	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)

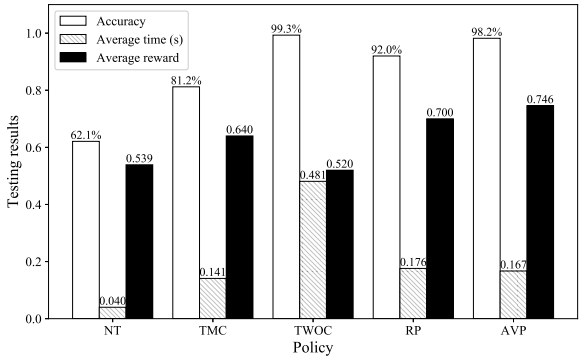

Fig. 8. Testing results with different policies under the linear form reward function when $\beta_{\text{lin}} = 0.0019$.

TABLE X
STATE–ACTION TABLE UNDER THE LOGARITHMIC FORM REWARD FUNCTION WHEN $\beta_{\log} = 0.001$

A_i \ $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 36)	(1, 36)	(1, 36)	(1, 30)	(1, 30)
η_2	(1, 36)	(1, 36)	(1, 36)	(1, 36)	(1, 36)
η_3	(1, 38)	(1, 38)	(1, 38)	(1, 38)	(1, 38)
η_4	(1, 42)	(1, 40)	(1, 36)	(1, 36)	(1, 36)
η_5	(0, /)	(0, /)	(1, 44)	(1, 44)	(1, 44)

TABLE XI
STATE–ACTION TABLE UNDER THE LOGARITHMIC FORM REWARD FUNCTION WHEN $\beta_{\log} = 0.0015$

A_i \ $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 36)	(1, 36)	(1, 36)	(1, 30)	(1, 30)
η_2	(1, 36)	(1, 36)	(1, 36)	(1, 36)	(1, 36)
η_3	(1, 38)	(1, 38)	(1, 38)	(1, 38)	(1, 38)
η_4	(0, /)	(0, /)	(1, 40)	(1, 36)	(1, 36)
η_5	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)

TABLE XII
STATE–ACTION TABLE UNDER THE LOGARITHMIC FORM REWARD FUNCTION WHEN $\beta_{\log} = 0.0017$

A_i \ $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 36)	(1, 36)	(1, 36)	(1, 36)	(1, 36)
η_2	(1, 38)	(1, 36)	(1, 36)	(1, 36)	(1, 36)
η_3	(1, 42)	(1, 38)	(1, 38)	(1, 38)	(1, 38)
η_4	(0, /)	(0, /)	(0, /)	(0, /)	(1, 38)
η_5	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)

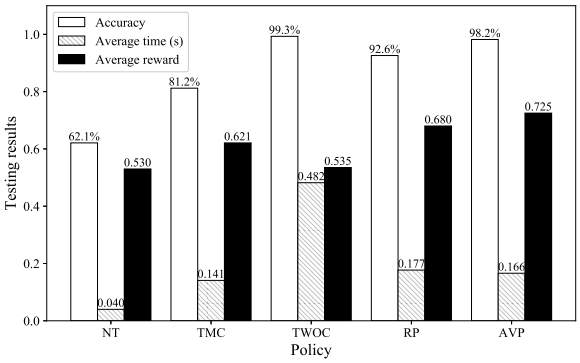

Fig. 9. Testing results with different policies under the logarithmic form reward function when $\beta_{\log} = 0.0017$.

3) Comparison of the Three Reward Functions: From Figs. 7–9, we can find that when using the policies NT, TMC, RP, and AVP, the process time T ranges from 0 to 400 ms; and when using the policy TWOC, the process time T is greater than 400 ms. In Fig. 4, it is seen that when $\beta_{\exp} = 0.0014$, $\beta_{\text{lin}} = 0.0019$, $\beta_{\log} = 0.0017$, and $T \in [0, 400]$ ms, the order of three reward functions sorted from the least to the greatest is logarithmic, linear, and exponential; and when $T > 400$ ms, the order of three reward functions reverses. Correspondingly, as shown in Figs. 7–9, for NT, TMC, RP, and AVP policies, the logarithmic reward function obtains the lowest average reward, the linear reward function obtains the second lowest, and the exponential reward function obtains the highest. But when using the policy TWOC, the order of the three reward functions reverses. It is also noted that the accuracy and average time do not follow the above order, which indicates that there is no unanimously optimal reward function of the three.

TABLE XIII
STATE–ACTION TABLE UNDER THE EXPONENTIAL FORM REWARD FUNCTION WHEN $E[h(t)]$ REDUCES BY 3 dB AND $N = 25\,000$

A_i \ $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 40)	(1, 40)	(1, 40)	(1, 40)	(1, 36)
η_2	(1, 38)	(1, 38)	(1, 38)	(1, 38)	(1, 38)
η_3	(1, 38)	(1, 38)	(1, 38)	(1, 38)	(1, 38)
η_4	(0, /)	(0, /)	(0, /)	(0, /)	(1, 44)
η_5	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)

TABLE XIV
STATE–ACTION TABLE UNDER THE EXPONENTIAL FORM REWARD FUNCTION WHEN $E[h(t)]$ INCREASES BY 3 dB AND $N = 20\,000$

A_i \ $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 36)	(1, 36)	(1, 36)	(1, 36)	(1, 36)
η_2	(1, 36)	(1, 36)	(1, 36)	(1, 30)	(1, 30)
η_3	(1, 38)	(1, 38)	(1, 38)	(1, 38)	(1, 38)
η_4	(1, 42)	(1, 42)	(1, 42)	(1, 42)	(1, 38)
η_5	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)

C. Experiments of the ϵ-Greedy Algorithm

To test the ϵ-greedy algorithm, we design two experiments for the three forms of reward functions. In the first experiment, we change the distance d between the camera sensor and the BS to make $E[h(t)]$ reduce or increase by 3 dB as compared with that in Section IV-B, and retrain the agent by the ϵ-greedy algorithm, thus to study the proposed algorithm's convergence and performance in the changed situation. In the second experiment, we further let the camera sensor move continually according to a 2-D random walk mobility model, and test the performance of the algorithm in this nonstationary environment. In both experiments, we initialize the agent with a trained state–action table by Algorithm 1 in which we set the average channel gain $E[h(t)]$ the same as that in Section IV-B.

1) Circumstances When Channel Gain E[h(t)] Increases or Reduces by 3 dB: For the exponential form reward function, we set the parameter $\beta_{\exp} = 0.0014$ and initialize the state–action table as Table V. The exploration rate is set to $\epsilon = 0.8$. The experiments indicate that when $E[h(t)]$ reduces by 3 dB, the training process converges after $N = 25\,000$ iterations; and when $E[h(t)]$ increases by 3 dB, it takes $N = 20\,000$ iterations to converge. Tables XIII and XIV show the corresponding updated state–action tables. If we compare the results of Tables V and XIII, it can be seen that when $E[h(t)]$ reduces by 3 dB, which means that the wireless channel condition becomes worse, more selected actions tend not to transmit the image to the back end. In this situation, even when the agent selects to transmit, the image compression ratio tends to be greater, which means that the size of the transmitted image becomes smaller in order to save bandwidth resources. In contrast if we compare the results of Tables V and XIV, we can

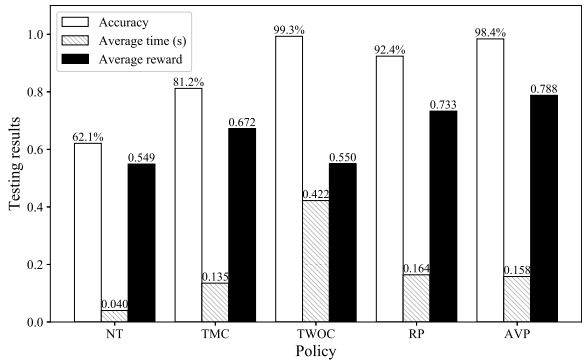

Fig. 10. Testing results with different policies under the exponential form reward function when $\beta_{\text{exp}} = 0.0014$ and $E[h(t)]$ increases by 3 dB.

TABLE XV
STATE–ACTION TABLE UNDER THE LINEAR FORM REWARD FUNCTION WHEN $E[h(t)]$ REDUCES BY 3 dB AND $N = 14\,000$

A_i \ $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 40)	(1, 40)	(1, 40)	(1, 40)	(1, 36)
η_2	(1, 38)	(1, 38)	(1, 38)	(1, 36)	(1, 36)
η_3	(1, 42)	(1, 42)	(1, 42)	(1, 38)	(1, 38)
η_4	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)
η_5	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)

see that when the wireless channel condition becomes better, i.e., $E[h(t)]$ increases by 3 dB, the situation reverses. There are more selected actions to transmit images to the back end, and the compression ratio tends to be smaller, which means that the quality of the transmitted images becomes better so that the MEC can produce more accurate recognition.

For the exponential form reward function in the new average channel conditions, we also compare the performance of AVP with NT, TMC, TWOC, and RP. Fig. 10 shows the testing results in terms of recognition accuracy, average time, and average reward from all tested 1600 image pairs, when $E[h(t)]$ increases by 3 dB. We can also obtain similar conclusions that AVP performs the best compared with other methods. If we compare the results of TMC, TWOC, RP, and AVP between Figs. 7 and 10, we can see that the cumulative rewards for these policies increase a little bit. This is because when channel condition becomes better, the agent either chooses to transmit more data (transmit more images or with smaller CRF values) to the back end, thus results in accuracy increase; or choose to transmit the same amount of data but with less transmission time. Thus, the system obtains better recognition accuracy, or less transmission time, or both. By (2) and (4), it can be deduced that the cumulative rewards become greater, the same as shown in Figs. 7 and 10.

Similarly, we do experiments to study the effects of the linear form reward function and the logarithmic form reward function in the new average channel conditions. For the linear form reward function, we set the parameter β_{lin} in (5) to 0.0019 and initialize the state–action table as Table VIII. For the logarithmic form reward function, we set the parameter β_{log} in (6) to 0.0017 and initialize the state–action table as Table XII. For both reward functions, the exploration rate ϵ is set to 0.8. Tables XV and XVI (Tables XVII and XVIII) show the results of updated state–action table under the linear (logarithmic) form reward function when the average channel gain reduces and increases by 3 dB, respectively. From the experimental results, we can see that despite the change of the average channel gain, we can still obtain a convergent state–action table after several iterations. Figs. 11 and 12 show the comparison between AVP and other policies under the linear and logarithmic form reward functions, respectively. Similar conclusions like the exponential form reward function can be also observed for the linear and logarithm form reward functions, which proves the effectiveness of the proposed ϵ-greedy algorithm.

TABLE XVI
STATE–ACTION TABLE UNDER THE LINEAR FORM REWARD FUNCTION WHEN $E[h(t)]$ INCREASES BY 3 dB AND $N = 21\,000$

A_i \ $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 36)	(1, 36)	(1, 36)	(1, 36)	(1, 36)
η_2	(1, 36)	(1, 36)	(1, 36)	(1, 36)	(1, 36)
η_3	(1, 38)	(1, 38)	(1, 38)	(1, 38)	(1, 38)
η_4	(0, /)	(1, 42)	(1, 36)	(1, 36)	(1, 36)
η_5	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)

TABLE XVII
STATE–ACTION TABLE UNDER THE LOGARITHMIC FORM REWARD FUNCTION WHEN $E[h(t)]$ REDUCES BY 3 dB AND $N = 24\,000$

A_i \ $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 40)	(1, 40)	(1, 40)	(1, 38)	(1, 36)
η_2	(1, 38)	(1, 38)	(1, 38)	(1, 36)	(1, 36)
η_3	(1, 42)	(1, 38)	(1, 38)	(1, 38)	(1, 38)
η_4	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)
η_5	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)

TABLE XVIII
STATE–ACTION TABLE UNDER THE LOGARITHMIC FORM REWARD FUNCTION WHEN $E[h(t)]$ INCREASES BY 3 dB AND $N = 24\,000$

A_i \ $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 36)	(1, 36)	(1, 36)	(1, 36)	(1, 36)
η_2	(1, 38)	(1, 36)	(1, 36)	(1, 36)	(1, 36)
η_3	(1, 38)	(1, 38)	(1, 38)	(1, 38)	(1, 38)
η_4	(0, /)	(0, /)	(0, /)	(0, /)	(1, 38)
η_5	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)

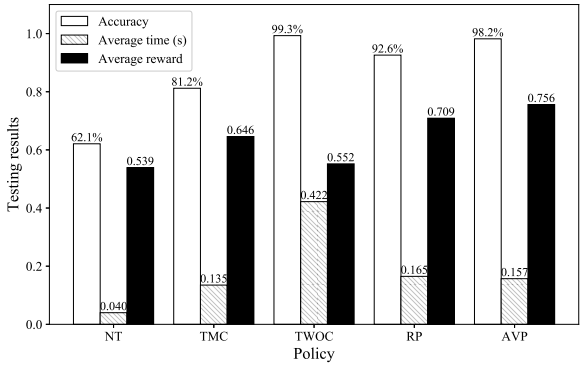

Fig. 11. Testing results with different policies under the linear form reward function when $\beta_{\text{lin}} = 0.0023$ and $E[h(t)]$ increases by 3 dB.

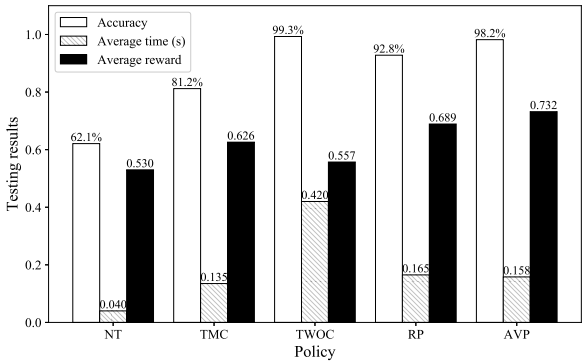

Fig. 12. Testing results under the logarithmic form reward function with different policies when $\beta_{\log} = 0.0017$ and $E[h(t)]$ increases by 3 dB.

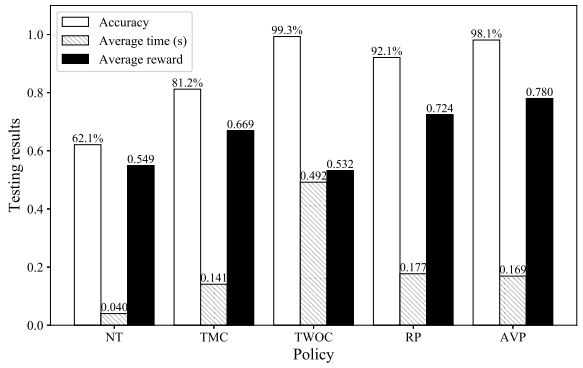

Fig. 13. Testing results under the exponential form reward function with different policies when the camera sensor moves randomly and $N = 25\,000$.

TABLE XIX
STATE–ACTION TABLE UNDER THE EXPONENTIAL FORM REWARD FUNCTION WHEN d IS TIME-VARYING AND $N = 25\,000$

A_i / $h(t)$ / η	h_1	h_2	h_3	h_4	h_5
η_1	(1, 40)	(1, 36)	(1, 36)	(1, 36)	(1, 36)
η_2	(1, 36)	(1, 38)	(1, 38)	(1, 36)	(1, 36)
η_3	(1, 38)	(1, 38)	(1, 38)	(1, 38)	(1, 38)
η_4	(0, /)	(0, /)	(1, 40)	(1, 42)	(1, 42)
η_5	(0, /)	(0, /)	(0, /)	(0, /)	(0, /)

2) Circumstances When Channel Gain E[h(t)] Changes Overtime Due to the Camera Sensor's Random Movement: To study the performance of the ϵ-greedy algorithm in a nonstationary environment, we deliberately vary the distance d between the camera sensor and the BS according to a 2-D random walk mobility model [37], thus the average channel gain $E[h(t)]$ changes overtime accordingly. Specifically, we suppose that the camera sensor moves in a circle with radius 50 m and the BS is located in its center. We first initialize $d = 30$ m and then make the camera sensor move by a random direction θ and speed v, which are drawn from a uniform distribution with the range $\theta \in [0, 2\pi)$ and $v \in (0, 5]$ m/s, respectively. We change the direction and speed of the camera sensor every 10 s. We further limit the camera sensor within the circle by reverting it directly if it hits the edge of the circle. Other experiment settings are the same as in Section IV-C1.

Table XIX shows the corresponding updated state–action table of the exponential form reward function after 25 000 iterations. It can be seen that the table is not stable because of some abnormal state–action pairs. For example, at the first and second columns of the second row, we can see that a worse channel condition results in a smaller CRF. It is not surprising because no convergence of the action-value table is guaranteed in the nonstationary environment when the distance between the camera sensor and the BS varies with the camera sensor's random movement. Although the action-value table is not converged, we can still achieve better average reward by using the proposed algorithm. Fig. 13 shows the testing results in terms of recognition accuracy, average time, and average reward of the NT, TMC, TWOC, RP, and AVP policies. It can be seen that AVP with the ϵ-greedy algorithm obtains the highest average reward compared with other policies.

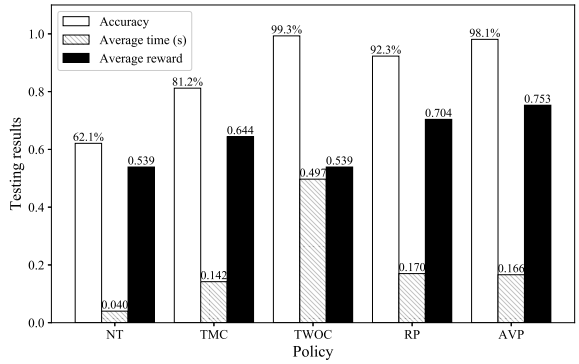

Fig. 14. Testing results under the linear form reward function with different policies when the camera sensor moves randomly and $N = 25\,000$.

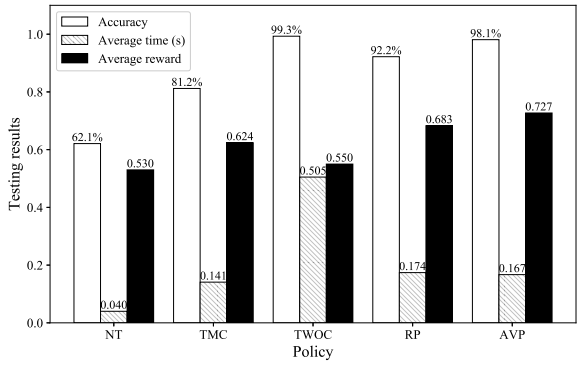

Fig. 15. Testing results under the logarithmic form reward function with different policies when the camera sensor moves randomly and $N = 25\,000$.

We further do similar experiments by using the linear and logarithmic form reward functions. Given iteration number is equal to 25 000, we obtain nonconverged state–action tables similar with Table XIX, which are omitted here for saving space. Figs. 14 and 15 show the comparison of testing results among different policies with the derived nonconverged state–action tables for both reward functions, respectively. It can be seen again that under either reward function AVP performs the best in average reward among all policies, showing the advantage of the ϵ-greedy algorithm in adapting to the nonstationary environment.

V. Conclusion

In this article, we have studied how to design an intelligent face recognition system based on mobile edge networks aiming at achieving high recognition accuracy and low time complexity. Specifically, we have designed image recognition algorithms for both the camera sensor and MEC server, and proposed the action-value approach to train actions of the system by jointly optimizing offloading decision and image compression parameters. Furthermore, we have introduced the ϵ-greedy method to ensure the agent can be adapted to the changing wireless communication channel. The experimental results not only show the advantages of the proposed system model in convergence capability and adaption to both channel conditions and recognition accuracies but also confirm its performance in terms of improving recognition accuracy or reducing recognition time overhead. Our design provides an example of how to balance the pros and cons of exploiting more distributed computation resources for a practical video surveillance application with clear application-level requirements and how to control the impact of lossy and rate constrained wireless links on the video surveillance application. Our future work will be carried out in two directions.

1) Designing algorithms to speed up the convergence of the proposed learning method. Generally, there are more sample efficient algorithms for the multiarmed bandit problem, such as Thompson sampling [38] and Bayesian estimation [39]. Recently, many algorithms are proposed in deep reinforcement learning to speed up system convergence [40], [41], which can also be implemented in our algorithm.
2) Studying the system-level design to allocate both computation and communication resources of different network elements among multiple coexisting video surveillance tasks.

References

[1] H. Hu *et al.*, "Intelligent video surveillance based on mobile edge networks," in *Proc. IEEE Int. Conf. Commun. Syst. (ICCS)*, Chengdu, China, Dec. 2018, pp. 286–291.

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

[3] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing: A key technology towards 5G," Sophia Antipolis, France, ETSI, White Paper, Sep. 2015.

[4] E. Eriksson, G. Dán, and V. Fodor, "Coordinating distributed algorithms for feature extraction offloading in multi-camera visual sensor networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 11, pp. 3288–3299, Nov. 2018.

[5] V. Young *et al.*, "Mobile-edge computing (MEC): Service scenarios," Sophia Antipolis, France, ETSI, White Paper, Nov. 2015.

[6] Y. Sun, X. Wang, and X. Tang, "Deeply learned face representations are sparse, selective and robust," in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR)*, Oct. 2015, pp. 2892–2900.

[7] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2016, pp. 499–515.

[8] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "SphereFace: Deep hypersphere embedding for face recognition," in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6738–6746.

[9] G. Ananthanarayanan *et al.*, "Real-time video analytics: The killer app for edge computing," *IEEE Comput.*, vol. 50, no. 10, pp. 58–67, Oct. 2017.

[10] C. Long, Y. Cao, T. Jiang, and Q. Zhang, "Edge computing framework for cooperative video processing in multimedia IoT systems," *IEEE Trans. Multimedia*, vol. 20, no. 5, pp. 1126–1139, May 2018.

[11] A. Anjum, T. Abdullah, M. F. Tariq, Y. Baltaci, and N. Antonopoulos, "Video stream analysis in clouds: An object detection and classification framework for high performance video analytics," *IEEE Trans. Cloud Comput.*, vol. 7, no. 4, pp. 1152–1167, Oct.–Dec. 2019.

[12] L. Xiao, Y. Li, X. Huang, and X. Du, "Cloud-based malware detection game for mobile devices with offloading," *IEEE Trans. Mobile Comput.*, vol. 16, no. 10, pp. 2742–2750, Oct. 2017.

[13] A. C. Caputo, *Digital Video Surveilliance and Security*, Oxford, U.K.: Butterworth-Heinemann Elsevier Ltd., 2010.

[14] "Cisco visual networking index: Forecast and trends, 2017–2022," Cisco, San Jose, CA, USA, Rep. C11-741490-00, 2018.

[15] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018.

[16] M. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 1991, pp. 586–591.

[17] M. Lades *et al.*, "Distortion invariant object recognition in the dynamic link architecture," *IEEE Trans. Comput.*, vol. 42, no. 3, pp. 300–311, Mar. 1993.

[18] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face description with local binary patterns: Application to face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 12, pp. 2037–2041, Oct. 2006.

[19] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," in *Proc. Workshop Real Life Images Detect. Alignment Recognit.*, Oct. 2008, pp. 1–11.

[20] Y. Sun, X. Wang, and X. Tang, "Deep learning face representation from 10 000 classes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Sep. 2014, pp. 1891–1898.

[21] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "DeepFace: Closing the gap to human-level performance in face verification," in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR)*, Sep. 2014, pp. 1701–1708.

[22] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 815–823.

[23] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Jun. 2016, pp. 507–516.

[24] S. Chen, Y. Liu, X. Gao, and Z. Han, "MobileFaceNets: Efficient CNNs for accurate real-time face verification on mobile devices," Apr. 2018. [Online]. Available: arXiv:1804.07573.

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

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

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

[28] K. Ogawa, K. Kanai, M. Takeuchi, J. Katto, and T. Tsuda, "Edge-centric field monitoring system for energy-efficient and network-friendly field sensing," in *Proc. IEEE Consum. Commun. Netw. Conf. (CCNC)*, Mar. 2018, pp. 1–6.

[29] A. R. Elias, N. Golubovic, C. Krintz, and R. Wolski, "Where's the bear? Automating wildlife image processing using IoT and edge cloud systems," in *Proc. IEEE Internet Things Design Implement. (IoTDI)*, Jun. 2017, pp. 247–258.

[30] D. Huang, C. Shan, M. Ardabilian, Y. Wang, and L. Chen, "Local binary patterns and its application to facial image analysis: A survey," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 41, no. 6, pp. 765–781, Mar. 2011.

[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[32] H. Trevor, T. Robert, and F. JH, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York, NY, USA: Springer, 2009.

[33] R. Sutton and A. Barto, *Reinforcement Learning, An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[34] S. Whiteson and P. Stone, "Evolutionary function approximation for reinforcement learning," *J. Mach. Learn. Res.*, vol. 7, pp. 877–917, May 2006.
[35] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multi-task cascaded convolutional networks," *IEEE Signal Process. Lett.*, vol. 23, no. 10, pp. 1499–1503, Aug. 2016.
[36] S. Guharoy and N. B. Mehta, "Joint evaluation of channel feedback schemes, rate adaptation, and scheduling in OFDMA downlinks with feedback delays," *IEEE Trans. Veh. Technol.*, vol. 64, no. 4, pp. 1719–1731, May 2013.
[37] R. R. Roy, *Handbook of Mobile Ad Hoc Networks for Mobility Models*. New York, NY, USA: Springer, 2011.
[38] S. Agrawal and N. Goyal, "Thompson sampling for contextual bandits with linear payoffs," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Atlanta, GA, USA, Jun. 2013, pp. 127–135.
[39] D. N. Hill, H. Nassif, Y. Liu, A. Iyer, and S. V. N. Vishwanathan, "An efficient bandit algorithm for realtime multivariate optimization," in *Proc. ACM Int. Conf. Knowl. Disc. Data Min. (KDD)*, Aug. 2017, pp. 1813–1821.
[40] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.
[41] K. Chua, R. Calandra, R. McAllister, and S. Levine, "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, Dec. 2018, pp. 4759–4770.

Haoji (Roland) Hu (Member, IEEE) received the B.Sc. degree in electrical engineering from Tsinghua University, Beijing, China, in 2002, and the Ph.D. degree in audio–visual person identification from the University of Southampton, Southampton, U.K., in 2007.

From 2007 to 2009, he was a Postdoctoral Researcher of digital watermarking with the Communications and Remote Sensing Laboratory, Universit Catholique de Louvain, Ottignies-Louvain-la-Neuve, Belgium. In 2009, he joined the Department of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China, as an Assistant Professor. He has been promoted to an Associate Professorship since 2013. He has published over 40 research papers in international journals and conference proceedings, e.g., *Pattern Recognition*, the IEEE TRANSACTIONS ON IMAGE PROCESSING, AAAI, and CVPR. He is the holder of seven national patents. His research interests mainly include image processing, computer vision, and deep learning.

Dr. Hu has obtained several Chinese National Scientific Projects.

Hangguan Shan (Member, IEEE) received the B.Sc. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 2004, and the Ph.D. degree in electrical engineering from Fudan University, Shanghai, China, in 2009.

From 2009 to 2010, he was a Postdoctoral Research Fellow with the University of Waterloo, Waterloo, ON, Canada. Since 2011, he has been with the College of Information Science and Electronic Engineering, Zhejiang University, where he is currently an Associate Professor. He is also with the Zhejiang Provincial Key Laboratory of Information Processing and Communication Networks, and SUTD-ZJU IDEA, Zhejiang University. His current research interests include cross-layer protocol design, resource allocation, and the quality-of-service provisioning in wireless networks.

Dr. Shan has co-received the Best Industry Paper Award from the 2011 IEEE WCNC held in Quintana Roo, Mexico. He has served as a Technical Program Committee Member of various international conferences, including the IEEE Global Communications Conference, the IEEE International Conference on Communications, the IEEE Wireless Communications and Networking Conference, and the IEEE Vehicular Technology Conference (VTC). He currently serves as the Track Co-Chair for Machine Learning and AI for Communications track of the IEEE VTC2020-Fall. He is currently an Editor of the IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING.

Chuankun Wang received the B.S. degree in information engineering from Central China Normal University, Wuhan, China in 2017. He is currently pursuing the master's degree with the Department of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China.

His research interests include computer vision, reinforcement learning, and deep learning face recognition.

Tengxu Sun received the B.S. degree in metallurgical engineering from Northeastern University, Qinhuangdao, China, in 2018. He is currently pursuing the M.S. degree in electronics and communication engineering with Zhejiang University, Zhejiang, China.

His research interests include computer vision, reinforcement learning, and model compression.

Xiaojian Zhen received the B.S. degree in information engineering and the M.S. degree in information and communication engineering from Zhejiang University, Hangzhou, China, in 2016 and 2019, respectively.

His research interests include resource allocation, LTE-U network, and stochastic geometry.

Kunpeng Yang received the B.S. degree in information engineering from Zhejiang University, Hangzhou, China, in 2018, where he is currently pursuing the M.S. degree in electronics and communication engineering.

His research interests include reinforcement learning and deep learning.

Lu Yu (Member, IEEE) received the B.Eng. degree in radio engineering and the Ph.D. degree in communication and systems from Zhejiang University, Hangzhou, China, in 1991 and 1996, respectively.

She has been a Professor with the Institute of Information and Communication Engineering, Zhejiang University, since 2003. She was a Senior Visiting Scholar with University Hannover, Hanover, Germany, in 2002 supported by China Scholarship Council and German Research Foundation (DFG). She was a Senior Visiting Scholar with the Chinese University of Hong Kong supported by the United College Resident Fellow Scheme in 2004. Her research area is video coding, multimedia communication, and relative ASIC design.

Prof. Yu was an Area Editor of *EURASIP Journal Signal Processing: Image Communication*. She is the General Chair of the Picture Coding Symposium (PCS) 2019, and also the International Steering Committee Member of PCS. She was the Chair of the Video-Subgroup of Audio Video Coding Standard of China from 2005 to 2017 and the Membership and Election Subcommittee of Technical Committee of Visual Signal Processing and Communication. She organized the Packet Video Workshop 2006 as the General Chair and IEEE Workshop of Multimedia Signal Processing 2011 as the Local Chair. She currently acts as the Video Subgroup Chair of ISO/IEC JTC1 SC29 WG11, which is known as MPEG and the elected Chair of the Technical Committee of Education and Outreach of IEEE Society of Circuits and Systems.

Zhaoyang Zhang (Member, IEEE) received the Ph.D. degree from Zhejiang University, Hangzhou, China, in 1998.

He is currently a Qiushi Distinguished Professor with Zhejiang University. He has authored or coauthored more than 300 peer-reviewed international journal and conference papers, including six conference best papers. His current research interests focus on the fundamental aspects of wireless communications and networking, such as information theory and coding, network signal processing and distributed learning, AI-empowered communications and networking, synergistic sensing, communication, and computation.

Dr. Zhang was a recipient of the National Natural Science Fund for Outstanding Young Scholars by NSFC in 2017. He was the General Chair, the TPC Co-Chair, or the Symposium Co-Chair for WCSP 2013/2018, Globecom 2014 Wireless Communications Symposium, and VTC2017-Spring Workshop HMWC. He is an Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, the IEEE TRANSACTIONS ON COMMUNICATIONS, and *IET Communications*.

Tony Q. S. Quek (Fellow, IEEE) received the B.E. and M.E. degrees in electrical and electronics engineering from the Tokyo Institute of Technology, Tokyo, Japan, in 1998 and 2000, respectively, and the Ph.D. degree in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 2008.

He is currently the Cheng Tsang Man Chair Professor with the Singapore University of Technology and Design (SUTD), Singapore. He also serves as the Acting Head of ISTD Pillar, the Sector Lead of the SUTD AI Program, and the Deputy Director of the SUTD-ZJU IDEA, Zhejiang University. His current research topics include wireless communications and networking, network intelligence, Internet of Things, URLLC, and big data processing.

Dr. Quek was honored with the 2008 Philip Yeo Prize for Outstanding Achievement in Research, the 2012 IEEE William R. Bennett Prize, the 2015 SUTD Outstanding Education Awards—Excellence in Research, the 2016 IEEE Signal Processing Society Young Author Best Paper Award, the 2017 CTTC Early Achievement Award, the 2017 IEEE ComSoc AP Outstanding Paper Award, and the 2016–2019 Clarivate Analytics Highly Cited Researcher. He has been actively involved in organizing and chairing sessions, and has served as a member of the Technical Program Committee as well as symposium chairs in a number of international conferences. He is currently serving as an Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, the Chair of IEEE VTS Technical Committee on Deep Learning for Wireless Communications, as well as an Elected Member of the IEEE Signal Processing Society SPCOM Technical Committee. He was an Executive Editorial Committee Member of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS and an Editor of the IEEE TRANSACTIONS ON COMMUNICATIONS and the IEEE WIRELESS COMMUNICATIONS LETTERS. He is a Distinguished Lecturer of the IEEE Communications Society.