Integrated Sensing, Computation, and Communication: System Framework and Performance Optimization

Yinghui He, *Graduate Student Member, IEEE*, Guanding Yu, *Senior Member, IEEE*, Yunlong Cai, *Senior Member, IEEE*, and Haiyan Luo

***Abstract*—Integrated sensing, computation, and communication (ISCC) has been recently considered as a promising technique for beyond 5G systems. In ISCC systems, the competition for communication and computation resources between sensing tasks for ambient intelligence and computation tasks from mobile devices becomes an increasingly challenging issue. To address it, we first propose an efficient sensing framework with a novel action detection module. In this module, a threshold is used for detecting whether the sensing target is static and thus the overhead can be reduced. Subsequently, we mathematically analyze the sensing performance of the proposed framework and theoretically prove its effectiveness with the help of the sampling theorem. Based on sensing performance models, we formulate a sensing performance maximization problem while guaranteeing the quality-of-service (QoS) requirements of tasks. To solve it, we propose an optimal resource allocation strategy, in which the minimum resource is allocated to computation tasks, and the rest is devoted to the sensing task. Besides, a threshold selection policy is derived and the results further demonstrate the necessity of the proposed sensing framework. Finally, a real-world test of action recognition tasks based on USRP B210 is conducted to verify the sensing performance analysis. Extensive experiments demonstrate the performance improvement of our proposal by comparing it with some benchmark schemes.**

***Index Terms*—Integrated sensing and communication, mobile edge computing, resource allocation, action recognition.**

I. Introduction

The past decades have witnessed innovations in network architecture and wireless technology for better communication services, such as high data rate, massive device access, and low latency. On the other hand, with artificial intelligence (AI) showing its power in almost every field, the intelligent revolution has finally come to wireless networks [1]. Implementing AI in wireless networks can provide diversified intelligent services in cellular networks [2]. One representation among them is to achieve ambient intelligence and smart environments [3] based on a large amount of real-time sensing data. Compared with vision-based sensing, radio-frequency (RF)-based sensing is a more appealing approach for data collection since it can work in non-line-of-sight scenarios, as well as it is light-needless [4] and privacy-preserving [5]. Three key aspects are necessary for enabling RF-based sensing in cellular networks, i.e., spectrum resource, hardware, and computation resource. The former two are used for transmitting and receiving sensing signals, and the last one is used for running sensing algorithms.

Y. He, G. Yu, and Y. Cai are with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China, and also with Zhejiang Provincial Key Laboratory of Information Processing, Communication and Networking (IPCAN), Hangzhou 310027, China (e-mail: 2014hyh@zju.edu.cn; yuguanding@zju.edu.cn; ylcai@zju.edu.cn).

H. Luo is with Lenovo Research, Shanghai 201210, China (e-mail: luohy7@lenovo.com).

Regarding the first and second aspects, directly allocating extra spectrum and equipping individual hardware would incur high overhead. To address this issue, integrated sensing and communication (ISAC) has been recently proposed since sensing and communication systems have a similar hardware architecture and can work on the same spectrum [6]. ISAC is expected to improve the spectral and energy efficiencies with a low hardware cost by jointly designing the sensing and communication functions [7]. To provide computation resource, a novel architecture, referred to mobile edge computing (MEC), has been recently recognized as a core technology by deploying edge servers with sufficient computation capacity at cellular base stations (BSs) [8]. By offloading the computation task to nearby edge servers, end-to-end latency can be significantly reduced as compared to cloud computing [9]. Motivated by this, integrated sensing, computation, and communication (ISCC) has been recently proposed by combining ISAC with MEC to achieve the goal of supporting ambient intelligence and smart environments [10], [11].

A. Related Work

ISCC is one of the key technologies for the next generation wireless networks and it will play a vital role in a wide range of application scenarios, such as smart home scenarios [10], internet of vehicle scenarios [12]–[14], and so on. ISAC and MEC are fundamental technologies for ISCC. The studies of ISAC have two stages, i.e., spectrum sharing and hardware sharing. For spectrum sharing, radar devices and communication devices share the same spectrum, and the main goal is to improve spectrum efficiency by jointly designing beamforming matrices [15], [16]. For instance, a joint design of transmit covariance matrix and radar sampling scheme has been proposed in [15] to reduce the effective interference power at the radar receiver. Moreover, since the sensing and communication systems have a similar hardware design, hardware sharing becomes possible to reduce the overhead. One way to realize hardware sharing is to enable the sensing function within a communication system [17]–[19]. For example, the work in [17] utilized a commercial WiFi card with a modified driver to realize the localization function. In addition, the other method is to design a dual-functional waveform that further exploits the potential of ISAC [20]–[23]. For instance, the authors of [20] proposed an optimal waveform toward a trade-off between radar and communication performance.

However, the above works only focused on the processes of transmitting and receiving sensing signals and have not thoroughly studied the computation process of sensing data. Note that for delay-sensitive sensing tasks, the computational delay is non-negligible. Meanwhile, offloading sensing data to cloud servers certainly incurs high communication latency. MEC can be adopted to address this issue since it provides computation ability to the edge network by equipping BSs with edge servers. The studies of MEC mainly concentrate on joint communication and computation resource allocation for minimizing the total energy consumption [24], [25] or end-to-end delay [26], [27].

Nevertheless, the design of resource allocation in MEC has not considered sensing tasks and their features. As a result, a major issue for ISCC is how to allocate the limited available resources, taking into account both communication and computation. Existing works [10], [11], [28], [29] have proposed their solutions for different scenarios to address this problem. The authors in [28] considered a general sensing task and jointly optimized computation resource allocation and MIMO precoding for radar waveforms and communication symbols with the delay constraint and resource limitation in ISCC systems. To reduce the interference between sensing and communication, intelligent reflecting surface (IRS) is adopted in [29] to suppress the interference between sensing and communication. A joint resource allocation and beamforming algorithm is proposed to minimize energy consumption. Differently, literatures [10], [11] investigated AI-based sensing tasks. Specifically, the training process of human motion recognition tasks in a federated edge learning (FEEL) system has been investigated in [10], and the resource allocation problem of ISCC was addressed to accelerate the convergence of FEEL. On the other hand, the authors of [11] studied the inference process of AI-based sensing tasks where the sensing accuracy is measured by an approximate but tractable metric, i.e., discriminant gain. An optimal joint transmit power and communication resource allocation strategy has been proposed for maximizing the discriminant gain. However, directly implementing AI-based sensing algorithms will bring a high computational expense in ISCC systems since they are computation-intensive. Meanwhile, in most scenarios, such as smart home scenarios, the sensing target might be static for an extended period[1], or there may even be no sensing target. Besides, existing works about RF-based sensing mainly focus on generalization issue [30]–[32] and privacy issue [5], [33], [34]. The former tries to improve the generalization ability of the sensing algorithm for automatically adapting to new and previously unseen sensing targets and environments, since RF signals are sensitive to the environments and sensing targets. The latter one aims to avoid the information leakage, since RF signals are non-intrusive and can be transmitted through the wall. Nevertheless, the computation overhead issue of sensing tasks has not been studied before. Inspired by this, we aim to propose a novel sensing framework to further improve computational and communication efficiencies while guaranteeing sensing performance.

B. Main Contributions

In this paper, we consider an ISCC system where a triple-functional BS equipped with an edge server needs to perform a sensing task, i.e., action recognition, for the ambient intelligence and also provide communication and computing services in the cellular network. Specifically, the sensing task serves the smart home scenarios where variations of wireless signals could be employed to detect or recognize human daily activities and thereby the BS can support diversified intelligent services, such as fall detection and intelligent home control [6]. Since the sensing and computation tasks compete for the limited computation and communication resources, it is crucial to design an effective framework and resource allocation strategy. This work is mostly related to the prominent ISAC work [42]. It only focuses on the tradeoff between recognition accuracy and communication rate. Moreover, a convolutional neural network (CNN)-based sensing algorithm was directly used in [42] for sensing tasks without considering the computation resource waste for the static state. Motivated by this, we aim to design an effective sensing framework and the corresponding resource allocation strategy to further improve the sensing, computation, and communication performance simultaneously. In particular, we address three main challenges: 1) how to design this sensing framework; 2) how to analyze the sensing performance based on the proposed sensing framework; 3) how to jointly allocate resources for maximizing the sensing performance under the quality-of-service (QoS) requirements of devices. The main contributions of this work are summarized as follows.

- We propose a novel and effective sensing framework where a key action detection module is placed before the CNN module. In the action detection module, the power of high-frequency components of sensing signals is compared with a threshold for determining whether the sensing target is static due to the fact that the action of the target will cause the variations of sensing signals.
- We quantitatively analyze the overall sensing performance of the proposed sensing framework, i.e., the sensing accuracy and delay. Specifically, the sampling theorem is applied to calculate the miss rate and false positive rate for the action detection module. The accuracy is modeled as a monotonically increasing function, and the delay of the CNN module is proportional to the sampling rate. Furthermore, we prove that our proposed sensing framework can save unnecessary computation resource, and the gain is positively related to the probability of the occurrence of the static state.
- A sensing accuracy maximization problem is formulated with the delay requirement of computation tasks. By addressing this problem, we propose an optimal resource allocation strategy in which the resource allocated to

[1]The static state means that the sensing scenario remains unchanged. For example, a human (i.e., the sensing target) keeps the same posture, which would not cause the variations of sensing signals over time.

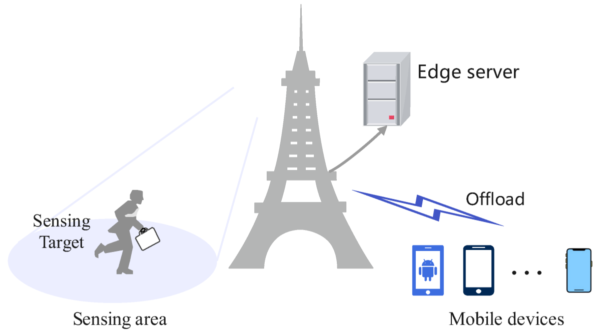


Fig. 1. An ISCC system.

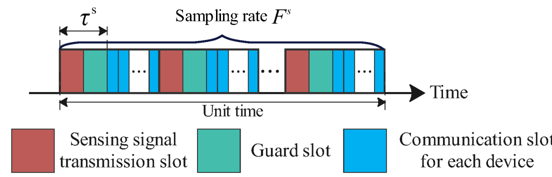


Fig. 2. Illustration of TDMA method at the triple-functional BS.

computation tasks is minimized and the remaining resource is allocated to the sensing task to maximize the accuracy. Besides, an optimal threshold selection policy is also derived.

- We conduct a real-world test of the action recognition task using the USRP B210. The proposed mathematical models are validated with the collected dataset. Furthermore, numerical results demonstrate that the proposed sensing framework and algorithm outperform several other benchmark schemes.

The rest of the paper is organized as follows. Section II introduces the ISCC system model, the computation task model, and the sensing framework. Section III analyzes the sensing performance of the proposed framework and proves its superiority. Section IV formulates a sensing performance maximization problem and decomposes it into two subproblems for developing an overall algorithm. Section V presents the test results, and the whole paper is concluded in Section VI.

II. System Model and Sensing Framework

In this section, we introduce the ISCC system, the computation task model, and the proposed effective sensing framework.

A. ISCC System

As depicted in Fig. 1, we consider an ISCC system containing one triple-functional BS equipped with an edge server and N mobile devices. The BS aims to conduct a sensing task and provide communication and computing services. Each device has its own computation task. Due to the limited computation resource at the mobile device, all computation tasks need to be offloaded to the edge server located at the BS via the wireless links. Meanwhile, the BS needs to perform a sensing task, i.e., an action recognition task. For this purpose, the wireless sensing signal is transmitted towards the sensing area at the sampling rate F^{s}. The echo is collected at the triple-functional BS and then fed back to the edge server for action recognition. To sum up, the BS has three functions: 1) communication function: mobile devices can offload their tasks to the BS via wireless links; 2) sensing function: the BS can transmit sensing signals; 3) computation function: the computation and sensing tasks can be completed with the help of the edge server.

In this work, since the interference between sensing and communication would seriously affect the sensing performance, the BS adopts the time division multiple access (TDMA) method to perform sensing and communication functions, as shown in Fig. 2.[2] Specifically, there are three types of time slots: 1) sensing signal transmission slot for transmitting the sensing signal; 2) guard slot for collecting sensing echoes; 3) communication slot for data offloading. Let τ^{s} denote the total proportion of one sensing signal transmission slot and one guard slot, i.e., total duration per unit time. Moreover, the BS transmits sensing signals at a sampling rate of F^{s}, which means there are F^{s} sensing signal transmission slots and F^{s} guard slots per unit time. Thus, the total proportion of communication resource occupied by the sensing task is $\tau^{\mathrm{s}}F^{\mathrm{s}}$. The proportion of communication resource allocated to device n for offloading per unit time is denoted by τ_n^{c}.[3] Therefore, we have $\tau^{\mathrm{s}}F^{\mathrm{s}} + \sum_{n=1}^{N}\tau_n^{\mathrm{c}} \leq 1$.[4]

B. Computation Task Model

The computation task of device n can be represented by a tuple $(V_n, C_n, T_n^{\max})$, where V_n (in bit) denotes the data size of the task, C_n (in CPU cycle/bit) denotes the computation intensity, i.e., the required number of CPU cycles for computing one bit, and $T_n^{\max}$ (in s) denotes the delay limitation. The computation task offloading of each device can be divided into two main phases: 1) *transmission phase*, where each device transmits its task to the BS with the allocated time slots; 2) *computation phase*, where the edge server computes the task. Note that the phase of returning the computation result can be neglected because of the small data size [24].

In the transmission phase, let B denote the system bandwidth and let σ_{c}^2 denote the channel noise power. Then, the instantaneous data rate of device n is given by

$$R_n = B\log_2\left(1+\frac{|h_n|^2 p_n}{\sigma_{\mathrm{c}}^2}\right), \tag{1}$$

where h_n is the channel gain between device n and the BS, and p_n is the transmit power of device n. Meanwhile, the proportion of communication resource occupied by device n is τ_n^{c}, that is, the time slot available for transmission of device n is τ_n per unit time. Thus, for device n, the transmitted data size per unit time is $\tau_n^{\mathrm{c}}R_n$, which means that the average data rate of device n is $\tau_n^{\mathrm{c}}R_n$. Thus, for device n, the delay of transmission phase can be expressed as

$$T_n^{\text{tran}} = \frac{V_n}{\tau_n^{\mathrm{c}}R_n}. \tag{2}$$

[2] We should note that the whole system is under the control of the BS and each device only can transmit data within the allocated time slots.

[3] The delay of computation tasks is not related to the sampling rate, and thus we directly describe the proportion of communication resource allocated to device n as τ_n^{c}.

[4] Compared with the duration of the time slot for sensing and communication tasks, the duration of the controlling signal is much shorter and thus can be ignored. This assumption has been widely adopted in similar works [10], [11].

In the computation phase, let f_n (in CPU cycle/s, i.e., Hz) denote the computation resource of the edge server allocated to device n. Then, according to [35], the delay of computation phase for device n can be expressed as

$$T_n^{\text{comp}} = \frac{V_n C_n}{f_n}. \tag{3}$$

Based on the above, the overall delay for device n to complete its task is given by

$$T_n = T_n^{\text{tran}} + T_n^{\text{comp}} = \frac{V_n}{\tau_n^{\text{c}} R_n} + \frac{V_n C_n}{f_n}. \tag{4}$$

C. Sensing Framework

For the sensing task of action recognition, the overall sensing framework contains two phases, i.e., *signal collection phase* at the BS and *action recognition phase* at the edge server, as shown in Fig. 3. The former one is utilized for collecting the radar echo signal, and the latter one is utilized for processing the signal to obtain the action type.[5]

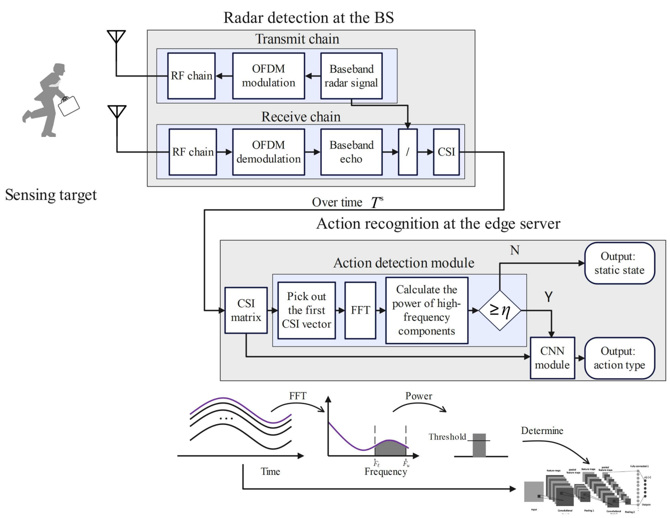


Fig. 3. The proposed sensing framework.

For the *signal collection phase* at the BS, we consider an orthogonal frequency division multiplexing (OFDM) radar [36] because it can be easily implemented in existing BSs.[6] With the OFDM radar, the total system bandwidth B is divided into K subcarriers. The baseband radar signal, denoted by x_k for subcarrier k, is transmitted via an RF chain after OFDM modulation. After reflection and OFDM demodulation, the baseband echo, denoted by y_k for subcarrier k, is obtained. Finally, the processed OFDM radar signal for subcarrier k can be calculated by $g_k = \frac{y_k}{x_k}$, which can be regarded as the channel state information (CSI) between the transmitting and receiving antennas of the BS. To detect the action type of sensing target, we need to collect the CSI samples over a period of time, whose length is denoted by T^{s}. With the sampling rate being F^{s}, the total collected CSI samples can construct a CSI matrix, denoted by $\boldsymbol{G} = [\boldsymbol{g}_1^H \ \boldsymbol{g}_2^H \cdots \boldsymbol{g}_K^H]^H \in \mathbb{C}^{K \times (T^{\text{s}} F^{\text{s}})}$, where $\boldsymbol{g}_k = [g_k[1] \ g_k[2] \cdots g_k[T^{\text{s}} F^{\text{s}}]] \in \mathbb{C}^{1 \times (T^{\text{s}} F^{\text{s}})}$ is the collected CSI vector for subcarrier k over the period of T^{s}, $g_k[m]$ is the m-th CSI sample for subcarrier k, and $(\cdot)^H$ denotes the operation of conjugate transpose.

For the *action recognition phase* at the edge server, in the conventional scheme, the collected CSI matrix is directly input into the CNN module [39]. However, in some scenarios (especially smart home scenarios), the target may not exist in the sensing area, or the sensing target may be in a static state. Thus, continuously employing CNN would result in high cost of computation resource. Meanwhile, we should note the fact that an action of a sensing target causes the CSI to vary over time, which leads to an increase in the power of high-frequency components. Motivated by this, we design an action detection module placed before the CNN module, as shown in Fig. 3. It is used for detecting whether the sensing target is in a static state or an action state. To achieve it, we first pick the CSI vector of the first subcarrier among the K subcarriers.[7] Then, to obtain the power of high-frequency components, we need to convert the CSI vector from the time domain to the frequency domain. Thus, we adopt the fast Fourier transform (FFT) to obtain the discrete Fourier transform (DFT) of the CSI vector, as

$$D[l] = \frac{1}{\sqrt{T^{\text{s}} F^{\text{s}}}} \sum_{m=1}^{T^{\text{s}} F^{\text{s}}} g_1[m] \exp\left(\frac{-j2\pi l m}{T^{\text{s}} F^{\text{s}}}\right). \tag{5}$$

Next, we need to calculate the power of high-frequency components. Let $[F_\ell, F_u]$ denote the range of high frequency and then the corresponding range for DFT $D[l]$ is $[\lfloor F_\ell T^{\text{s}} \rfloor, \lceil F_u T^{\text{s}} \rceil]$ according to [40]. The power of high-frequency components can be calculated as the ratio of the total energy in the range $[F_\ell T^{\text{s}}, F_u T^{\text{s}}]$ to the number of samples, as

$$P = \frac{1}{T^{\text{s}} F^{\text{s}}} \sum_{l=\lfloor F_\ell T^{\text{s}} \rfloor}^{\lceil F_u T^{\text{s}} \rceil} |D[l]|^2. \tag{6}$$

Finally, by comparing the power P with a pre-set threshold, denote by η, we can determine whether the sensing target is in the static state or the action state. If it is in the action state, the CSI matrix $\boldsymbol{G}$ is further input into the CNN module for action recognition; otherwise, the result of the static state is output.

The key to the proposed action detection module is the threshold. If the threshold is set too high, it may cause a large number of action state instances to be recognized as the static state, which pulls down the average recognition accuracy. If the threshold is set too low, a number of static state instances may be detected as the action state, increasing both the cost of computation resource and delay. Therefore, it is important to select an appropriate threshold to reduce both computation consumption and delay without sacrificing

[5] In this paper, we adopt the CNN-based sensing algorithm since it is one of the most popular solutions for the RF-based action recognition task [37]. Moreover, our proposed method can also be extended to other sensing algorithms.

[6] Our design can be extended to other radar types, such as frequency-modulated continuous wave (FMCW) radar [38].

[7] Here, we take the first subcarrier as an example. In fact, any of the K subcarriers can be used for the action detection module. Moreover, all K subcarriers can be used in the proposed action detection module with the help of a voting classifier, which deserves future work.

accuracy. Moreover, the sampling rate also influences the accuracy. With a higher sampling rate, it can intuitively be seen that the accuracy becomes higher since more information about the sensing target is collected. In the next section, we will analyze the effect of threshold and sampling rate on the average accuracy and delay of the proposed sensing framework.

III. Sensing Performance Analysis

In this section, we first analyze the sensing accuracy and delay of our proposed sensing framework and then prove its effectiveness.

A. Sensing Accuracy

In our proposed sensing framework, the sensing accuracy is jointly determined by two modules, i.e., the action detection module and the CNN module. We assume that the action recognition task has I recognition types and their set is denoted by $\{1, 2, \cdots, I\}$, where the first type is the static state and $\{2, 3, \cdots, I\}$ is the set of action state types.[8] For the action detection module, we can describe its performance in terms of miss rate and false positive rate. To be specific, the miss rate for the i-th action type is defined as the proportion of the i-th action instances that are detected as the static state, i.e.,

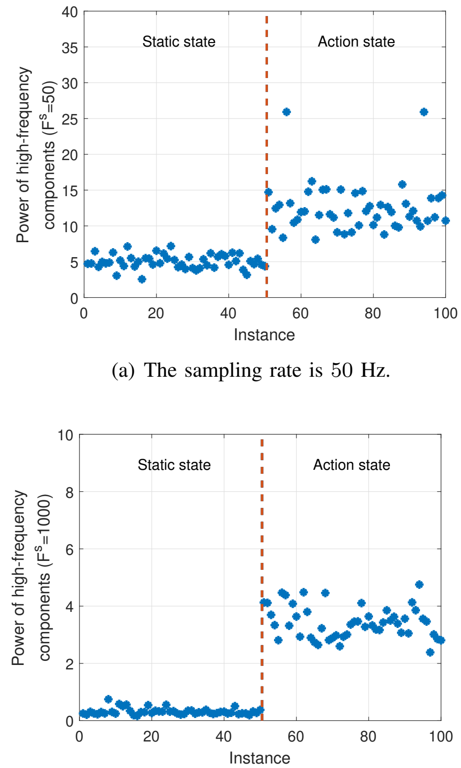
$$p_i^o = \frac{M_i^0}{M_i}, \ i = 2, 3, \cdots, I, \tag{7}$$

where M_i is the total number of the i-th action type instances and M_i^0 is the number of the i-th action type instances that are recognized as the static state. The false positive rate is defined as the proportion of the static state instances that are recognized as the action state, i.e.,

$$p^l = \frac{M_0^{\mathrm{a}}}{M_0}, \tag{8}$$

where M_0 is the total number of static state instances and M_0^{a} is the number of static state instances that are recognized as the action state. For the CNN module, recognition accuracy is utilized to describe the performance, which is defined as the ratio of the correctly recognized instances to the total number of recognition instances. Although we have defined performance metrics, they cannot be used for theoretical analysis since they are based on statistical results. Thus, in the following, we will theoretically model them with the help of the sampling theorem.

First of all, we focus on the action detection module. As we mentioned before, the key is the threshold. Moreover, the sampling rate also affects p_i^o and p^l. To illustrate the effects of the threshold and sampling rate, we collect some instances and plot the power of high-frequency components in Fig. 4. From the figure, we can observe that the power of the action state is obviously higher than that of the static state, which demonstrates the effectiveness of the proposed action detection module. Besides, as the sampling rate increases, the power gap between the two states becomes higher, which further reduces both the miss rate and the false positive rate. It is intuitively reasonable since the sensing target becomes more informative, and it is easier to distinguish between the static state and the action state with a higher sampling rate. Recall that we aim to optimize the sensing performance in this paper. The challenge, however, is that the mathematical relationships among the sampling rate, threshold, miss rate, and false positive rate are unclear.



(a) The sampling rate is 50 Hz.

(b) The sampling rate is 1000 Hz.

Fig. 4. The power of high-frequency components for 100 instances under different sampling rates. Specifically, for the first 50 instances, the target is in the static state. For the last 50 instances, the target is in the action state. The test setting and dataset are introduced in Section V-A.

To model the effect of sampling rate, we need to consider the continuous-time CSI of the first subcarrier in the time domain, denoted by $g_1^c(t)$. Then, the collected CSI can be rewritten as the sample of the continuous-time CSI at sampling rate F^{s}, as

$$g_1[m] = g_1^{\mathrm{c}}(t)|_{t=m/F^{\mathrm{s}}} + w_0[m] \tag{9}$$

$$= g_1^{\mathrm{c}}\left(\frac{m}{F^{\mathrm{s}}}\right) + w_0[m], \ m = 1, 2, \cdots, T^{\mathrm{s}}F^{\mathrm{s}}, \tag{10}$$

where $w_0[m]$ is the estimation error that can be modeled as the Gaussian noise with mean being zero and variance being σ^2. Then, according to the sampling theorem [40], the relationship between the DFT $D[l]$ of $g_1[m]$ and the Fourier transform of $g_1^{\mathrm{c}}(t)$, denoted by $D^{\mathrm{c}}(F)$, can be expressed as

$$D[l] = \frac{1}{\sqrt{T^{\mathrm{s}}F^{\mathrm{s}}}} \left(\sum_{m=0}^{T^{\mathrm{s}}F^{\mathrm{s}}-1} g_1^{\mathrm{c}}\left(\frac{m}{F^{\mathrm{s}}}\right) \exp\left(\frac{-j2\pi l m}{T^{\mathrm{s}}F^{\mathrm{s}}}\right) \tag{11}$$

[8]The number of types, i.e., I, should be given in the prior design and training processes for AI-based sensing tasks. The value of I is generally determined according to specific scenarios and tasks. In the test part, we consider 8 typical action types.

$$+\sum_{m=0}^{T^{\mathrm{s}}F^{\mathrm{s}}-1} w_0[m]\exp\left(\frac{-j2\pi lm}{T^{\mathrm{s}}F^{\mathrm{s}}}\right)\Bigg)$$

$$=\sqrt{\frac{F^{\mathrm{s}}}{T^{\mathrm{s}}}}\sum_{m=-\infty}^{+\infty} D^{\mathrm{c}}\left(\left(\frac{l}{T^{\mathrm{s}}F^{\mathrm{s}}}-m\right)F^{\mathrm{s}}\right)+W_0[l], \quad (12)$$

where $W_0[l]$ is the equivalent Gaussian noise in the frequency domain with zero mean and variance σ^2. Supposing $D^{\mathrm{c}}(F)=0$ when $|F|>F^{\mathrm{s}}/2$, we can further simplify (12) as

$$D[l]=\sqrt{\frac{F^{\mathrm{s}}}{T^{\mathrm{s}}}}D^{\mathrm{c}}\left(\frac{l}{T^{\mathrm{s}}}\right)+W_0[l]. \quad (13)$$

Then, the power calculated in the proposed action detection module can be rewritten as

$$P=\frac{1}{T^{\mathrm{s}}F^{\mathrm{s}}}\sum_{l=\lfloor F_\ell T^{\mathrm{s}}\rfloor}^{\lceil F_u T^{\mathrm{s}}\rceil}\left|\sqrt{\frac{F^{\mathrm{s}}}{T^{\mathrm{s}}}}D^{\mathrm{c}}\left(\frac{l}{T^{\mathrm{s}}}\right)+W_0[l]\right|^2 \quad (14)$$

$$=\frac{1}{(T^{\mathrm{s}})^2}\sum_{l=\lfloor F_\ell T^{\mathrm{s}}\rfloor}^{\lceil F_u T^{\mathrm{s}}\rceil}\left|D^{\mathrm{c}}\left(\frac{l}{T^{\mathrm{s}}}\right)+\sqrt{\frac{T^{\mathrm{s}}}{F^{\mathrm{s}}}}W_0[l]\right|^2. \quad (15)$$

Thus, P follows the noncentral chi-squared distribution with the degree of freedom (DoF) of $\lceil F_u T^{\mathrm{s}}\rceil-\lfloor F_\ell T^{\mathrm{s}}\rfloor+1$. Since the high frequency range is wide enough and T^{s} is long enough, e.g., $F_\ell=10$ Hz and $T^{\mathrm{s}}=3$ s, P approximately follows the normal distribution according to [41] and its mean and variance are given by

$$\mu_P=\lambda+\frac{r}{T^{\mathrm{s}}F^{\mathrm{s}}},\ \sigma_P^2=4\frac{\sigma^2}{T^{\mathrm{s}}F^{\mathrm{s}}}\lambda+2\frac{\sigma^2}{(T^{\mathrm{s}}F^{\mathrm{s}})^2}r, \quad (16)$$

respectively, where

$$\lambda=\frac{1}{(T^{\mathrm{s}})^2}\sum_{l=\lfloor F_\ell T^{\mathrm{s}}\rfloor}^{\lceil F_u T^{\mathrm{s}}\rceil}\left|D^{\mathrm{c}}\left(\frac{l}{T^{\mathrm{s}}}\right)\right|^2,$$
$$r=\sigma^2\left(\lceil F_u T^{\mathrm{s}}\rceil-\lfloor F_\ell T^{\mathrm{s}}\rfloor+1\right). \quad (17)$$

Moreover, instances of the same action cannot be exactly the same. For example, the target may not stand at the same location when the target performs the same action. Therefore, we add a constant, denoted by σ_d^2, into the variance and it describes the distinction of action instances. Therefore, the revised variance is given by

$$\sigma_P^2=4\frac{\sigma^2}{T^{\mathrm{s}}F^{\mathrm{s}}}\lambda+2\frac{\sigma^2}{(T^{\mathrm{s}}F^{\mathrm{s}})^2}r+\sigma_d^2. \quad (18)$$

Based on the above analysis, we can mathematically model the relationship among the sampling rate, threshold, miss rate, and false positive rate, as shown in the following proposition.

Proposition 1: The miss rate for the i-th action is given by

$$p_i^o=Q\left(\frac{\mu_{P,i}-\eta}{\sigma_{P,i}}\right), i=2,3,\cdots,I, \quad (19)$$

and the false positive rate for the static state is given by

$$p^l=Q\left(\frac{\eta-\mu_{P,1}}{\sigma_{P,1}}\right), \quad (20)$$

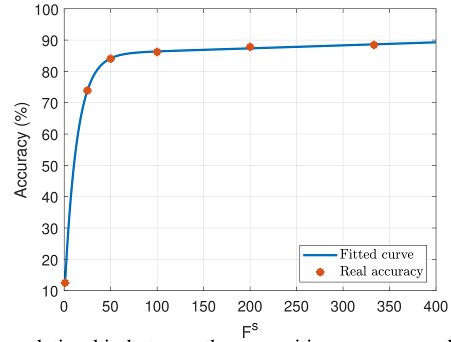


Fig. 5. The relationship between the recognition accuracy and sampling rate.

where $Q(\cdot)$ is the Q-function[9], $\mu_{P,i}=\lambda_i+\frac{r_i}{T^{\mathrm{s}}F^{\mathrm{s}}}$, and $\sigma_{P,i}^2=4\frac{\sigma^2}{T^{\mathrm{s}}F^{\mathrm{s}}}\lambda_i+2\frac{\sigma^2}{(T^{\mathrm{s}}F^{\mathrm{s}})^2}r_i+\sigma_{d,i}^2$. λ_i, r_i, and $\sigma_{d,i}^2$ are parameters that can be obtained via fitting on the collected sensing dataset.

From Proposition 1, we can find that p_i^o increases but p^l decreases with the threshold, which confirms the statement we mentioned in Section II-C. Moreover, both the mean and variance decrease with sampling rate, which is consistent with Fig. 4.

Next, we need to model the accuracy of the CNN module. According to [42], the accuracy is positively correlated with the sampling rate. A higher sampling rate means more information about the sensing target, and it is reasonable that the CNN module can achieve higher accuracy. Thus, it can be represented by a monotonically increasing function, denoted by $\alpha(F^{\mathrm{s}})$. To verify this, we collect a sensing dataset and adopt one of the most well-known CNNs, ResNet50 [43]. After fully training until convergence, the relationship between the accuracy and sampling rate is shown in Fig. 5. We can observe that the accuracy is monotonically increasing with F^{s}.

Based on the above analysis of two modules, we can express the recognition accuracy of each type and show the average recognition accuracy with the probability distribution of recognition types. Specifically, the accuracy for the i-th action type is $(1-p_i^o)\,\alpha(F^{\mathrm{s}})$ with $i\geq 2$ since instances of the i-th type can be recognized correctly only after they are not detected as the static state by the action detection module. The accuracy for the static state is $(1-p^l)+p^l\alpha(F^{\mathrm{s}})$ where $(1-p^l)$ and $p^l\alpha(F^{\mathrm{s}})$ represent the probabilities that instances of the static state are correctly detected by the action detection module, and incorrectly detected by the action detection module but correctly detected by the CNN module, respectively. Therefore, the average recognition accuracy of the action recognition task is given by

$$A=\sum_{i=2}^{I}p_i^m\left(1-p_i^o\right)\alpha(F^{\mathrm{s}})+p_1^m\left(\left(1-p^l\right)+p^l\alpha(F^{\mathrm{s}})\right), \quad (21)$$

where $p_i^m=\frac{M_i}{\sum_{i=1}^{I}M_i}$ denotes the probability of the i-th type with $\sum_{i=1}^{I}p_i^m=1$.

B. Sensing Delay

The delay of the signal collection phase is constant, and the cost of the action detection module is small enough to

[9]https://cnx.org/contents/hDU5uzaA@2/The-Q-function

be neglected since the main computation cost in the action detection module is the FFT operation. Therefore, we focus on the CNN module.[10] Recall that the input of the CNN module is matrix $\boldsymbol{G}$ and its size is $K \times (T^{\text{s}} F^{\text{s}})$. Meanwhile, the main part of the CNN is the convolutional layer, which causes the major computation delay of the CNN. According to [44], the computational complexity of convolutional layers is proportional to the input matrix size, i.e., $K \times (T^{\text{s}} F^{\text{s}})$. Thus, the overall computational complexity of the CNN module is $\mathcal{O}\left(K T^{\text{s}} F^{\text{s}}\right)$. Let f^{s} denote the computation resource at the edge sever allocated to the CNN module. The corresponding delay can be modeled as

$$T^{\text{CNN}} = \frac{C^{\text{s}} K T^{\text{s}} F^{\text{s}}}{f^{\text{s}}}, \tag{22}$$

where C^{s} is the computation cost of the CNN module per element.

Since we have added the action detection module, not every instance would pass through the CNN module. Thus, we utilize the average delay to represent the performance of sensing tasks. Based on the analysis in Section III-A, the probability of an instance passing through the CNN module is

$$p^{\text{CNN}} = \sum_{i=2}^{I} p_i^m \left(1 - p_i^o\right) + p_1^m p^l, \tag{23}$$

where the first part is for action types and the second part is for the static state. Combining the probability in (23) with the delay in (22), the average delay of the sensing task is

$$T^{\text{s}} = \left(\sum_{i=2}^{I} p_i^m \left(1 - p_i^o\right) + p_1^m p^l \right) \frac{C^{\text{s}} K T^{\text{s}} F^{\text{s}}}{f^{\text{s}}}. \tag{24}$$

C. Performance Comparison

With the above analysis, we can identify the performance improvement of the proposed sensing framework by comparing it with the conventional one where there is no action detection module. The performance gain can be derived as shown in the following theorem.

Theorem 1: Under the same recognition accuracy target and delay requirement, the proposed sensing framework needs less computation resource than the conventional scheme when the following condition is satisfied

$$1 - \sum_{i=2}^{I} \frac{p_i^m \alpha(F^{\text{s}}) \sigma_{P,1}}{p_1^m \left(1 - \alpha(F^{\text{s}})\right) \sigma_{P,i}} \exp\left(\frac{(\mu_{P,1})^2}{2\sigma_{P,1}^2} - \frac{(\mu_{P,i})^2}{2\sigma_{P,i}^2} \right) > 0. \tag{25}$$

The performance gain, defined as the ratio of the saved computation resource to the required computation resource in the conventional framework, is

$$\rho = \min\left\{ \frac{p_1^m (1 - p^l)}{A} |_{A=\alpha(F^{\text{s}})}, 1 - p^{\text{CNN}} |_{\eta=\eta_u} \right\}, \tag{26}$$

where $\eta_u = \min\limits_{i \in \{2, \cdots, I\}} \mu_{P,i}$.

Proof: Please refer to Appendix A. ■

Remark 1: According to Theorem 1, we can verify the performance gain, that is, the proposed sensing framework can reduce the cost of computation resource without lowering the recognition accuracy. In other words, compared with the conventional framework, it can improve the accuracy with the same computation resource. To understand (25), we can focus on the probability distribution of action, i.e., $\{p_i^m\}$, and the accuracy of the CNN module, i.e., $\alpha(F^{\text{s}})$. As the probability of the static state p_1^m becomes higher, the condition is easier to be satisfied since the action detection module can avoid more unnecessary computation cost. Meanwhile, as the accuracy of the CNN module increases, the condition is more difficult to satisfy, because more instances would be input into the CNN module to achieve higher average recognition accuracy. Based on the above analysis, the proposed sensing framework is suitable for smart home scenarios, since the sensing target (e.g., human) keeps in the static state most of the time. When the condition is met, there are two cases. The performance gain is $1 - p^{\text{CNN}}|_{\eta=\eta_u}$ when the action detection module can almost perfectly distinguish between action instances and static instances, otherwise, it is $\frac{p_1^m(1-p^l)}{A}$. In the first case, the performance gain is only determined by the probability of passing through the CNN module, i.e., p^{CNN}. Furthermore, the relationship between them is negatively correlated since the computation cost increases with the probability of passing through the CNN module. In the second case, the performance gain is influenced by both the accuracy of the CNN module and the probability of the static state. The performance gain decreases because more instances are fed into the CNN module to achieve higher overall accuracy as the CNN accuracy increases. The performance gain increases with the probability of the static state. It is because, as the probability of the static state increases, the threshold in the action detection module can be set higher without pulling down the accuracy according to (21), and fewer instances are input into the CNN module. This eventually reduces the probability of computation. Besides, even in the worst case, the proposed sensing framework maintains the same performance as the conventional framework.

IV. Sensing Accuracy Maximization

In this section, we formulate a mathematical problem to maximize the sensing accuracy and then decompose it into two subproblems for achieving its solution.

A. Problem Formulation

Thus far, we have mathematically modeled the sensing performance including the average recognition accuracy in (21) and the average sensing delay in (24). In the following, we aim to solve the third challenge mentioned before, i.e., maximizing the sensing performance. To be specific, we aim to maximize the sensing accuracy under the delay requirement, denoted by $T^{\text{s,max}}$, by optimizing the threshold and sampling rate. Meanwhile, we need to consider the delay requirement of each computation task since the computation and communication resource is limited. Let f^{e} denote the total computation

[10]For example, according to test results, when F^s is 200 Hz and the computation resource of the edge server is 2.64 GHz, the computation delay of the action detection is around 2×10^{-5} s while that of the CNN module is around 0.75 s.

resource at the edge server. Then, we formulate the sensing accuracy maximization problem, as

$$\begin{aligned}
\max_{\{F^{\mathrm{s}},\eta,f^{\mathrm{s}},\tau_n^{\mathrm{c}},f_n\}} \quad & A=\sum_{i=2}^{I} p_i^m\left(1-p_i^o\right)\alpha(F^{\mathrm{s}}) \\
& +p_1^m\left((1-p^l)+p^l\alpha(F^{\mathrm{s}})\right), && \text{(27a)}\\
\text{s.t.} \quad & T^{\mathrm{s}}\le T^{\mathrm{s,max}}, && \text{(27b)}\\
& T_n\le T_n^{\max},\ \forall n, && \text{(27c)}\\
& \tau^{\mathrm{s}}F^{\mathrm{s}}+\sum_{n=1}^{N}\tau_n^{\mathrm{c}}\le 1, && \text{(27d)}\\
& f^{\mathrm{s}}+\sum_{n=1}^{N}f_n\le f^{\mathrm{e}}, && \text{(27e)}\\
& 0\le\eta\le\eta_u, && \text{(27f)}\\
& F^{\mathrm{s}}\in\mathcal{Z}^+, f^{\mathrm{s}},\tau_n^{\mathrm{c}},f_n\ge 0, && \text{(27g)}
\end{aligned}$$

where $\mathcal{Z}^+$ denotes the set of positive integers. In the above, (27b) and (27c) are the delay constraints for the sensing task and computation tasks, respectively, (27d) and (27e) are the communication and computation resource limitation constraints, respectively, and (27f) gives the range of the threshold where the upper bound is η_u that is the minimum power mean among all action types. The reason for setting an upper bound is that the accuracy would drop rapidly if the threshold is higher than η_u.

B. Problem Decomposition

By jointly considering (27a), (27b), and (27f), we can find that the effect of sampling rate on the sensing accuracy is quite complicated. Thus, we first give F^{s} and then solve the accuracy maximization problem (27) with a given F^{s}. After that, by performing an exhaustive search on F^{s}, we can find the optimal solution to problem (27). To solve the accuracy maximization problem (27) with given F^{s}, we can find that the sensing and computation tasks compete for the computation resource. Meanwhile, we aim to maximize the sensing accuracy. Therefore, we can minimize the computation resource required for the computation tasks while satisfying their delay requirement, and allocate all remaining resource to the sensing task. In this way, the accuracy maximization problem (27) for a given F^{s} can be intuitively decoupled into two subproblems. The first one is a resource allocation problem that minimizes the required computation resource by optimizing τ_n^{c} and f_n, as

$$\begin{aligned}
\min_{\{f_n,\tau_n^{\mathrm{c}}\}} \quad & \sum_{n=1}^{N} f_n, && \text{(28)}\\
\text{s.t.} \quad & \text{(27c), (27d), and (27g).}
\end{aligned}$$

The second one is a threshold selection problem that maximizes the sensing accuracy by optimizing the threshold with $f^{\mathrm{s}}=f^{\mathrm{e}}-\sum_{n=1}^{N}f_n$, as

$$\begin{aligned}
\min_{\eta} \quad & A, && \text{(29)}\\
\text{s.t.} \quad & \text{(27b) and (27f).}
\end{aligned}$$

In the following, we will solve the two subproblems in turn.

C. Resource Allocation Strategy

First of all, we can prove that problem (28) is convex, as shown in Lemma 1.

Lemma 1: Problem (28) is a convex optimization problem.

Proof: The objective function and constraints (27d) and (27g) are linear, and constraint (27c) is a linear combination of two convex functions, i.e., $1/\tau_n^{\mathrm{c}}$ and $1/f_n$. As a result, problem (28) is convex, which ends the proof. ∎

Based on Lemma 1, we utilize the Lagrangian method to solve problem (28) and the corresponding partial Lagrangian function can be expressed as

$$\mathcal{L}=\sum_{n=1}^{N}f_n+\sum_{n=1}^{N}\lambda_n\left(T_n-T_n^{\max}\right) +\mu\left(\tau^{\mathrm{s}}F^{\mathrm{s}}+\sum_{n=1}^{N}\tau_n^{\mathrm{c}}-1\right), \tag{30}$$

where λ_n and μ are the Lagrange multipliers associated with the constraints (27c) and (27d). Let $\{f_n^\star,\tau_n^{\mathrm{c},\star}\}$ denote the optimal solution to problem (28). Then, by applying the Karush-Kuhn-Tucker (KKT) conditions, we can derive the following theorem.

Theorem 2: The optimal solution to problem (28) can be given by

$$\begin{cases}
\tau_n^{\mathrm{c},\star}=\dfrac{V_n}{T_n^{\max}}\left(\dfrac{1}{R_n}+\sqrt{\dfrac{C_n}{\mu^\star R_n}}\right),\\
f_n^\star=\dfrac{V_n}{T_n^{\max}}\left(C_n+\sqrt{\dfrac{\mu^\star C_n}{R_n}}\right),
\end{cases} n=1,2,\cdots,N, \tag{31}$$

where $\mu^\star$ is the optimal Lagrange multiplier, as

$$\mu^\star=\left(\frac{\sum_{n=1}^{N}\frac{V_n}{T_n^{\max}}\sqrt{\frac{C_n}{R_n}}}{1-\tau^{\mathrm{s}}F^{\mathrm{s}}-\sum_{n=1}^{N}\frac{V_n}{T_n^{\max}R_n}}\right)^2. \tag{32}$$

Proof: Please refer to Appendix B. ∎

Remark 2: Theorem 2 gives the optimal resource allocation strategy. The allocated amount of resource to each computation task is negatively related to the data rate and is positively related to the computation intensity and task size. The computation resource decreases sublinearly with the data rate exponentially by 1/2. Moreover, we should note that the remaining computation resource is all allocated to the sensing task. Thus, the computation resource for the sensing task increases sublinearly with exponential 1/2 of the data rate.

D. Threshold Selection Policy

Now, we focus on problem (29). To solve it, we need to explore the properties of the objective function and constraint (27b), i.e., A and T^{s}. T^{s} decreases with $\eta\in[0,\eta_u]$ since p_i^o monotonically increases with η and p^l monotonically decreases with η. Thus, delay constraint (27b) is equivalent to $\eta\le\eta^T$, where η^T satisfies that $T^{\mathrm{s}}=T^{\mathrm{s,max}}$ and can be obtained via the exhaustive search algorithm. The computational complexity is $\mathcal{O}\left(\log(\eta_u/\epsilon)\right)$, where ϵ is the tolerance of accuracy.

As for A, it is a concave function with $\eta \in [\mu_{P,1}, \eta_u]$ since both p^l and p_i^o are convex with $\eta \in [\mu_{P,1}, \eta_u]$. However, when $\eta \in [0, \mu_{P,1}]$, A is not concave since p^l is not convex, and thus we cannot obtain the optimal solution directly. To address it, we adopt piecewise linear approximation (PLA) for p^l. First of all, the set $[0, \mu_{P,1}]$ is divided into M subsets and the m-th subset is $\left[\frac{(m-1)\mu_{P,1}}{M}, \frac{m\mu_{P,1}}{M}\right]$. In each subset, p^l is approximated as a linear function. Then, A is a concave function with η belonging to each subset since p^l is linear and p_i^o is convex. Let $\eta^{A,m}$ denote the threshold that maximizes A within the m-th subset and it can be obtained via the Golden-section search algorithm [46] with the computational complexity of $\mathcal{O}\left(\log(\mu_{P,1}/M/\epsilon)\right)$. Besides, the threshold that maximizes A within $[\mu_{P,1}, \eta_u]$ is denoted by $\eta^{A,M+1}$. Then, the threshold that maximizes A within $[0, \eta_u]$, denoted by η^A, is given by

$$\eta^A = \max_{m \in \{1, \cdots, M+1\}} \eta^{A,m}. \tag{33}$$

The computational complexity for calculating η^A is $\mathcal{O}\left(M \log(\mu_{P,1}/M/\epsilon) + \log((\eta_u - \mu_{P,1})/\epsilon) + M\right)$.

By jointly considering the objective function and delay constraint, there are two cases.

- When $\eta^A \geq \eta^T$, η^A is a feasible solution to problem (29) and the maximum accuracy is achieved when $\eta = \eta^A$.
- When $\eta^A < \eta^T$, the delay requirement cannot be satisfied when $\eta = \eta^A$ and the accuracy decreases with $\eta > \eta^A$. Thus, the maximum accuracy is achieved when $\eta = \eta^T$.

Combining the above two cases, the optimal solution to problem (29) is shown in the following theorem.

Theorem 3: The optimal threshold selection policy is given by

$$\eta^\star = \max\left(\eta^T, \eta^A\right). \tag{34}$$

Remark 3: From Theorem 3, the delay requirement may limit the sensing accuracy. We should note that when the accuracy is limited by the delay constraint in the proposed sensing framework, there is no feasible solution in the conventional framework. This is because the proposed sensing framework is equivalent to the conventional one when $\eta = 0$, and η^T should be set higher than 0 to meet the delay constraint as T^{s} decreases with $\eta \in [0, \eta_u]$. Furthermore, the accuracy of the proposed framework is higher than that of the conventional one when the accuracy is not limited by the delay constraint. As a result, the proposed framework performs better than the conventional one.

E. Overall Algorithm

Before presenting the overall algorithm, we give an upper bound of F^{s} in the following.

Lemma 2: An upper bound of F^{s} is given by

$$F_u^{\mathrm{s}} = \left\lfloor \frac{1}{\tau^s}\left(1 - \frac{N \min\limits_n V_n f^{\mathrm{e}} / \min\limits_n R_n}{\max\limits_n T_n^{\max} f^{\mathrm{e}} - N \min\limits_n V_n \min\limits_n C_n}\right)\right\rfloor. \tag{35}$$

Proof: Please refer to Appendix C. ∎

Based on the above analysis, we can obtain the overall algorithm to problem (27), as shown in Algorithm 1.

Algorithm 1: Overall Algorithm for Solving Problem (27).

Initialize the maximal error tolerance $\epsilon > 0$ and the maximal accuracy $A^\star = 0$;
for $F^s \in \{1, \cdots, F_u^s\}$ **do**
    % Resource allocation strategy;
    Obtain the optimal resource allocation strategy $\{\tau_n^{\mathrm{c},\star}, f_n^\star\}$ with Theorem 2;
    % Threshold Selection Policy;
    Obtain $\eta^\star$ with Theorem 3;
    Calculate the overall accuracy A;
    if $A^\star < A$ **then**
        $A^\star = A$;
    end
end

Specifically, we perform exhaustive searching on F^{s}. In each loop iteration, we solve problems (28) and (29) with Theorems 2 and 3, respectively. By comparing the accuracy under different F^{s}, we can find the maximal accuracy of problem (27), denoted by $A^\star$. The computational complexity of each loop iteration is $\mathcal{O}\left(N + M \log(\mu_{P,1}/M/\epsilon) + \log((\eta_u - \mu_{P,1})/\epsilon) + M\right)$, where the computational complexity for obtaining the optimal resource allocation strategy is $\mathcal{O}\left(N\right)$. Since the number of loop iterations is F_u^{s}, the total computational complexity of Algorithm 1 is

$$\mathcal{O}\left(F_u^{\mathrm{s}}\left(N + M \log(\mu_{P,1}/M/\epsilon) + \log(\eta_u/\epsilon) + M\right)\right). \tag{36}$$

Furthermore, Algorithm 1 is based on the exhaustive search method. Meanwhile, each loop solves the resource allocation problem (28) with Theorem 2 and the threshold selection problem (29) with Theorem 3. Thus, the convergence of Algorithm 1 can be guaranteed.

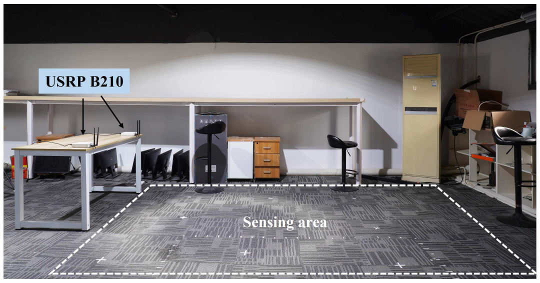
To tackle extreme scenarios where the computation ability of the BS is limited, we also introduce a low-complexity algorithm. Specifically, in each loop, we can set $M = 1$ for reducing the computational complexity of solving the threshold selection policy optimization problem to $\mathcal{O}\left(\log(\mu_{P,1}/\epsilon) + \log(\eta_u/\epsilon)\right)$. Regarding the number of loops, we observe the sensing accuracy generally increases with the sampling rate according to Fig. 12 in Section V-C. Thus, we heuristically seek to find the maximum sampling rate that ensures there is a feasible solution to problem (27) with binary search algorithm. As a result, the computational complexity of the low-complexity algorithm is $\mathcal{O}\left(\log(F_u^{\mathrm{s}})\left(N + \log(\mu_{P,1}/\epsilon) + \log(\eta_u/\epsilon)\right)\right)$.

V. Test Results

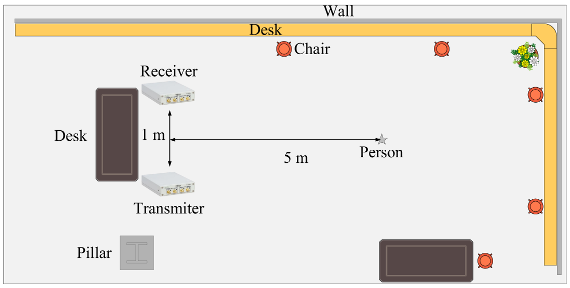
In this section, we conduct experiments to validate the effectiveness of the proposed algorithm.

A. Test Parameters

The test settings are provided as follows unless otherwise specified. We consider a triple-functional BS with a



(a) Real-world experiment scenario.



(b) Schematic diagram of the experiment scenario.

Fig. 6. The experiment setup for the sensing task.

radius of 300 m, and 15 devices are randomly located in the coverage. According to the long term evolution (LTE) standards [45], the system bandwidth is 4 MHz and the noise spectral density is -174 dBm/Hz. The channel gain between each device and the BS is generated by following the path loss model: $128.1 + 37.6 \log_{10}(d[km])$, where d denotes the distance between each device and the BS in kilometer and the small-scale fading is set to Rayleigh distributed with uniform variance. The transmit power of each device is set as 24 dBm. The total computation resource at the edge server is set as 40 GHz. For the computation task at the device, the data size and computation intensity follow the uniform distribution with $V_n \in [0.3, 1]$ Mbits and $C_n \in [400, 1000]$ CPU cycles/s, respectively, and the delay tolerance of all computation tasks is 0.4 s [35].

For the sensing task, we consider an OFDM radar signal, where the FFT number is 64 and the cyclic prefix length is 16. The time of one sensing signal transmission slot is 128 us. The experiment setup is shown in Fig. 6. Specifically, we use one USRP B210 to generate OFDM radar signals and one USRP B210 to collect radar echoes. Eight volunteers are invited to perform eight different behaviors, i.e., standing still, kicking, raising a hand, waving, bending down, walking, sitting down, and standing up, where standing still refers to the static state. We totally collect 3,200 CSI matrices as a dataset, where 2,560 CSI matrices are used for training and 640 CSI matrices are used for testing. We adopt ResNet50 for the CNN module. It is trained on a Linux server equipped with four NVIDIA GeForce GTX 3080 GPUs and tested on a Linux personal computer (PC) equipped with an Intel i7-8700K CPU, which is regarded as the edge server.

The detailed procedures of the simulation and experiment are presented in Fig. 7. Based on the sensing setting, we first collect the sensing dataset with USRP B210. The dataset is used for fitting λ_i, r_i, and $\sigma^2_{d,i}$ mentioned in Proposition 1

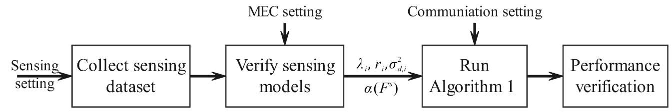


Fig. 7. The detailed procedures of the simulation and experiment.

and training the CNN module to obtain the accuracy function $\alpha(F^s)$. The results can verify the proposed sensing models. Next, under the communication setting, the performance of the proposed Algorithm 1 can be confirmed with λ_i, r_i, $\sigma^2_{d,i}$, and $\alpha(F^s)$. Note that when there is no feasible solution, the accuracy is set as 12.5% which is a lower bound of the action recognition task and is achieved with the random guess.

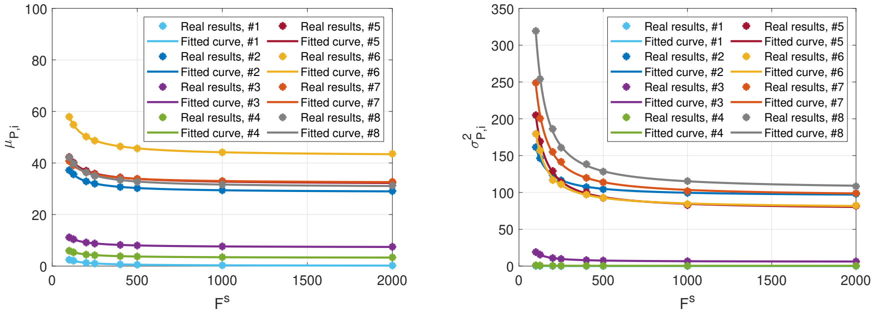
B. Model Verification

In Section III, we have proposed two mathematical models: one is a power model of high-frequency components, i.e., P, for describing the miss rate and false positive rate in the action detection module, and the other is a delay model for the CNN module. Here, we use the collected real-world sensing dataset to verify both models. First of all, Fig. 8 shows the mean and variance of P for eight types and the corresponding fitted curves, respectively. The real mean and variance of each type are calculated based on the collected CSI matrices, and the corresponding fitted curves are obtained by following (16) and (18), respectively. It can be observed that both the mean and variance decrease with the sampling rate and the fitted curves well match the real results under different sampling rates. Furthermore, we also plot Fig. 9 to verify Proposition 1 and describe the relationship among miss rate, false positive rate, sampling rate, and threshold. We only plot the miss rate of one action type due to the page limit. The remaining action types show similar results. It can be observed that the real-world results match the theoretical results well, which further proves the effectiveness of our proposed model for the action detection module.

In Fig. 10, we show the real-world computation delay of the CNN module with the Linux PC and the corresponding fitted curves based on (22). It can be seen that the computation delay increases with the sampling rate but decreases with the allocated computation resource. Moreover, the fitted curves match the real delay well. This verifies the effectiveness of the proposed delay model for the CNN module.

C. Algorithm Investigation and Performance Comparison

First of all, we show the convergence behavior of Algorithm 1 and the low-complexity algorithm in Fig. 11. Since Algorithm 1 is based on the exhaustive search algorithm, it requires more than 1,000 iteration loops when the error tolerance (i.e., the step between two adjacent sampling rates) is set as 1. To accelerate convergence, the tolerance can be set to 10, and the number of loops is about 100. The final accuracy with the error tolerance being 10 is almost the same as that with the error tolerance being 1. Moreover, Algorithm 1 can be performed in parallel. Specifically, in each loop of Algorithm 1, the overall accuracy of different sampling rates can be calculated in parallel. The edge server used in our test contains 6 CPU



(a) The fitted curves of the power mean for eight types.

(b) The fitted curves of the power variance for eight types.

Fig. 8. The verification of the mathematical model for the power of high-frequency components.

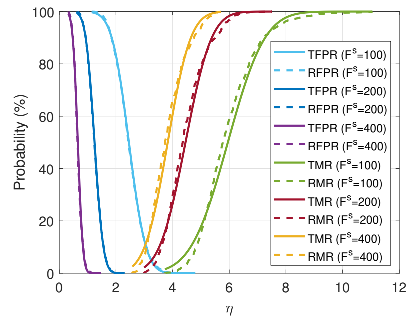


Fig. 9. The verification of miss rate and false positive rate models in Proposition 1. "TFPR" represents the theoretical false positive rate. "RFPR" represents the real-world false positive rate. "TMR" represents the theoretical miss rate. "RMR" represents the real-world miss rate.

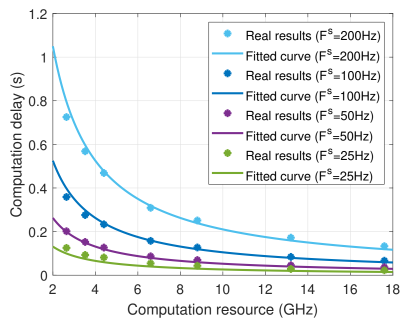
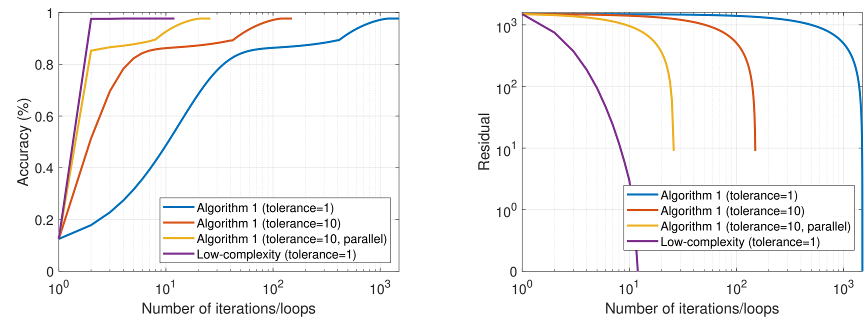


Fig. 10. The real-world computation delay of the CNN module with different sampling rates.



(a) Accuracy vs. number of iterations/loops.

(b) Residual vs. number of iterations/loops

Fig. 11. Convergence behavior of the proposed algorithms.

cores and the number of loops can be reduced to about 30. As for the low-complexity algorithm, it is based on the binary search algorithm and the number of iterations is about 10, which verifies its high efficiency. Besides, the final accuracy of the low-complexity algorithm is almost the same as that of Algorithm 1, which verifies the high performance of the low-complexity algorithm.

Next, we also plot the optimized sensing accuracy under the given sampling rate, as shown in Fig. 12. From the figure, the optimized sensing accuracy substantially increases with the sampling rate. However, when the sampling rate is too high, there is no feasible solution and the sensing accuracy is set as 12.5% as we have mentioned before. This is because that the delay requirement of the sensing task cannot be satisfied when the sampling rate is too high.

To show the performance advantage of the sensing framework and the proposed algorithm, we select the following benchmark algorithms.

- Conventional scheme. The action detection module is not adopted in this scheme and the CSI matrix is directly input into the CNN module [42]. The resource allocation strategy is similar to the proposed algorithm. To be specific, it can be obtained by setting the threshold as 0 in Algorithm 1.
- Average computation resource scheme. The computation resource is equally allocated to each device and sensing task, i.e., $f_n = \dfrac{f^{\rm e}}{N+1}, \forall n$ and $f^{\rm s} = \dfrac{f^{\rm e}}{N+1}$. The communication resource allocation strategy and the threshold selection policy are the same as those in the proposed algorithm.
- Average communication resource scheme. The communication resource is equally allocated to each device and sensing task, i.e., $\tau_n^{\rm c} = \dfrac{1}{N+1}, \forall n$ and $F^{\rm s} = \left\lfloor \dfrac{1}{(N+1)\tau^{\rm s}} \right\rfloor$. The computation resource allocation strategy and the threshold selection policy are the same as those in the proposed algorithm.

Fig. 13 shows the effect of the average computation resource at the edge server on the accuracy. From the figure, we can observe that our proposed scheme provides the best performance among four schemes. Specifically, when the overall accuracy is 90%, our proposed scheme requires 2.58 GHz average computation resource at the edge server, while the conventional scheme requires 3.19 GHz average computation resource. The proposed sensing framework saves about 19.1% computation resource, and this gap becomes more remarkable when accuracy is 95%. This verifies the effectiveness of the proposed sensing framework. When the computation resource is sufficient, the accuracy gap between the proposed scheme and the conventional scheme becomes very small. In this case, the accuracy of the CNN module is almost 100% and all instances are input into the CNN module. Moreover, compared with the average computation resource scheme, our proposed scheme achieves higher performance when the computation resource is insufficient and shows the similar performance when the computation resource is sufficient. It demonstrates the effectiveness of the proposed resource allocation strategy. Besides, the average communication resource scheme achieves

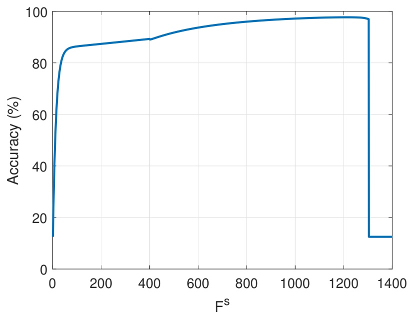


Fig. 12. The optimized accuracy under the given sampling rate.

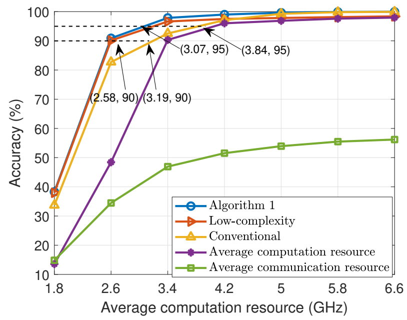


Fig. 13. Average computation resource vs. accuracy.

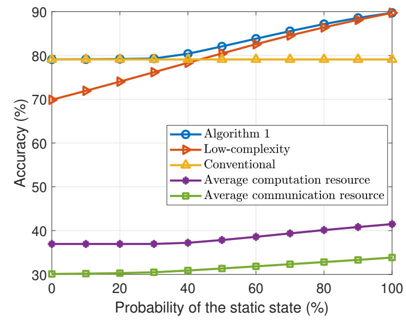


Fig. 14. Probability of the static state vs. accuracy.

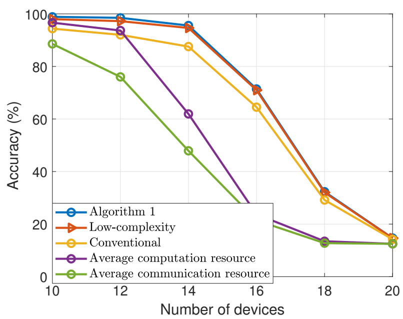


Fig. 15. The number of devices vs. accuracy.

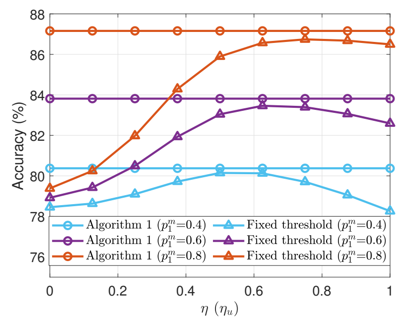


Fig. 16. The influence of the threshold.

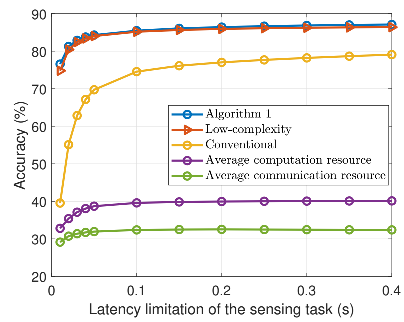


Fig. 17. Delay requirement of sensing task vs. accuracy.

much lower accuracy than the proposed scheme even if the computation resource is sufficient since its sampling rate is given and thus the accuracy is limited. Furthermore, the performance of the proposed low-complexity algorithm is close to that of Algorithm 1, which confirms that the low-complexity algorithm can be used in the extreme scenario.

Fig. 14 shows the effect of the probability of the static state on the accuracy. It can be seen that the proposed scheme offers the same accuracy as the conventional scheme when the probability is low. As we can imagine, the threshold should be set close to 0 when the probability of the static state is low; otherwise, a large number of action instances would be detected as the static state, which reduces the recognition accuracy. In this case, the impact of the action detection module on the ISCC system is small. As the probability of the static state increases, the threshold increases, and the action detection module shows its impact, i.e., reducing the required computation resource. Thus, more computation resource can be allocated to computation tasks, and then more communication resource can be saved for sensing. It eventually gives rise to an increase in sampling rate and accuracy improvement. In the same way, the accuracy of the average communication resource scheme and average computation resource scheme also increases with the probability of the static state. The above results demonstrate that the proposed sensing framework is applicable for the scenario with a high probability of the static state. This conclusion is consistent with Theorem 1, which proves that the performance gain brought by the proposed sensing framework is positively related to the probability of the static state. Moreover, the performance of the proposed low-complexity algorithm is worse than that of the conventional scheme when the probability of the static state is low. It is because the low-complexity is suboptimal and cannot be applied in this case.

Fig. 15 shows the effect of the number of devices on the accuracy. We can observe that the accuracy of the four analyzed schemes all decreases with the number of devices. It is because we need to meet the delay requirements of computation tasks, and thus the communication and computation resources are preferentially allocated to the devices. The remaining amount of resource decreases with the number of devices, which decreases the accuracy. When the number of devices is 10, the computation resource at the edge server is abundant, and thus the sampling rates of the four schemes are high enough, and the sensing accuracy approaches the upper limit, i.e., 100%. When the number of devices is 20, the computation resource at the edge server is limited and thus the sampling rates of the four schemes are low enough, and the sensing accuracy approaches the lower limit, i.e., 12.5%. Besides, the accuracy of the four schemes is almost the same when the number of devices is 10 and 20. In contrast, the accuracy of the proposed and conventional schemes drops more slowly than that of the average communication/computation resource scheme. It demonstrates that the proposed resource allocation strategy can suppress the effect of the number of devices.

To show the effectiveness of the threshold selection policy, we also compare the proposed scheme with the fixed threshold scheme under different probabilities, as shown in Fig. 16. To be specific, we consider that the ratio of threshold to its upper limit η_u is constant in the fixed threshold scheme. From the figure, our proposed scheme shows better performance since the proposed threshold selection policy is optimal, in which the threshold is adaptively selected based on the allocated resource and the probability. Moreover, the threshold that achieves the peak of the fixed threshold scheme increases with the probability of the static state. This is consistent with the explanation for Fig. 14, where the threshold increases with the probability of the static state for reducing the unnecessary computation resource cost.

Fig. 17 shows the effect of the delay requirement of the sensing task on the accuracy of the four analyzed schemes. From the figure, the impact of the delay requirement is tiny when the delay requirement becomes loose. This result is reasonable since the accuracy grows slowly when the sampling rate is higher than 50 Hz, as shown in Fig. 5. In this case, the decrease of the delay requirement may lead to a drop in the sampling rate but would not lead to a decline in the accuracy. As the delay requirement becomes strict, the sampling rate becomes lower than 50 Hz. In this case, the accuracy drops rapidly as the delay requirement decreases according to Fig. 5. Moreover, the performance of the proposed scheme drops more slowly than the conventional scheme, further demonstrating the proposed scheme's superiority.

VI. Conclusion

In this paper, we studied an ISCC system and developed an effective sensing framework with an action detection module to avoid unnecessary computation cost for the static state. The sampling theorem was adopted for analyzing the performance of the proposed sensing framework. Through analysis, we identified that the performance gain brought by our proposal is positively related to the probability of the static state. Furthermore, a sensing accuracy maximization problem was formulated with the delay constraints of the sensing task and communication tasks. After analyzing the structure, the original problem was decomposed into two subproblems. By solving them, we proposed an optimal resource allocation strategy and an optimal threshold selection policy and then developed an overall algorithm to maximize the accuracy. Finally, the analyses for the proposed sensing framework and the proposed algorithm were validated by a real-world test, which demonstrated the superiority of our proposals.

Appendix A
Proof of Theorem 1

First of all, we should note that the proposed sensing framework is the same as the conventional one when $\eta = 0$, i.e., $A = \alpha(F^{\mathrm{s}})$. To obtain the performance gain, we need to find that p^{CNN} should be less 1 with $A \geq \alpha(F^{\mathrm{s}})$. Since p^{CNN} decreases with η, it is equivalent to find an $\eta > 0$ that satisfies $A \geq \alpha(F^{\mathrm{s}})$. A sufficient condition is that A increases with η when η approaches 0, which means $\frac{\partial A}{\partial \eta}|_{\eta=0} > 0$. Thus, we can obtain the condition in (25) with simple mathematical calculations.

Next, we can calculate the performance gain in two different cases.

1) *Case 1:* If A is always higher than $\alpha(F^{\mathrm{s}})$ when $\eta \in (0, \eta_u]$, η should be η_u for obtaining the highest performance gain. Therefore, ρ is given by

$$\rho = 1 - p^{\mathrm{CNN}}|_{\eta=\eta_u}. \tag{37}$$

2) *Case 2:* If there is an η that satisfies $A = \alpha(F^{\mathrm{s}})$ when $\eta \in (0, \eta_u]$, we have the following equation:

$$\begin{aligned}\alpha(F^{\mathrm{s}}) = A = &\sum_{i=2}^{I} p_i^m \left(1 - p_i^o\right) \alpha(F^{\mathrm{s}}) \\ &+ p_1^m \left((1 - p^l) + p^l \alpha(F^{\mathrm{s}})\right).\end{aligned} \tag{38}$$

Then, ρ is given by

$$\begin{aligned}\rho &= 1 - p^{\mathrm{CNN}} = 1 - \left(\sum_{i=2}^{I} p_i^m \left(1 - p_i^o\right) + p_1^m p^l\right) \\ &= 1 - \frac{A - p_1^m (1 - p^l)}{\alpha(F^{\mathrm{s}})} = \frac{p_1^m (1 - p^l)}{A}.\end{aligned} \tag{39}$$

Combining both cases, the performance gain is given as shown in (26), which ends the proof.

Appendix B
Proof of Theorem 2

According to the KKT conditions, the necessary and sufficient conditions of the optimal solution can be expressed as

$$\frac{\partial \mathcal{L}}{\partial \tau_n^{\mathrm{c},\star}} = -\lambda_n^\star \frac{V_n}{\left(\tau_n^{\mathrm{c},\star}\right)^2 R_n} + \mu^\star = \begin{cases} = 0, & \tau_n^{\mathrm{c},\star} > 0, \\ \geq 0, & \tau_n^{\mathrm{c},\star} = 0, \end{cases} \tag{40}$$

$$\frac{\partial \mathcal{L}}{\partial f_n^\star} = 1 - \lambda_n^\star \frac{V_n C_n}{\left(f_n^\star\right)^2} = \begin{cases} = 0, & f_n^\star > 0, \\ \geq 0, & f_n^\star = 0, \end{cases} \tag{41}$$

$$\lambda_n^\star \left(T_n - T_n^{\max}\right) = 0, \quad \lambda^\star \geq 0, \quad n = 1, 2, \cdots, N, \tag{42}$$

$$\mu^\star \left(\tau^{\mathrm{s}} F^{\mathrm{s}} + \sum_{n=1}^{N} \tau_n^{c,\star} - 1\right) = 0, \quad \mu^\star \geq 0. \tag{43}$$

From constraint (27c), we can find that $\tau_n^{\mathrm{c},\star} > 0$ and $f_n^\star > 0$. Combining (40) and (41), we can derive the relationship between $f_n^\star$ and $\tau_n^{\mathrm{c},\star}$, as

$$f_n^\star = \tau_n^{\mathrm{c},\star} \sqrt{\mu^\star R_n C_n}, \; n = 1, 2, \cdots, N, \tag{44}$$

and we can find that $\mu^\star > 0$ and $\lambda_n^\star > 0$. Next, according to (42), we have

$$\tau_n^{\mathrm{c},\star} = \frac{V_n}{T_n^{\max}} \left(\frac{1}{R_n} + \sqrt{\frac{C_n}{\mu^\star R_n}}\right), \; n = 1, 2, \cdots, N, \tag{45}$$

where $\mu^\star$ satisfies $\tau^{\mathrm{s}} F^{\mathrm{s}} + \sum_{n=1}^{N} \tau_n^{\mathrm{c},\star} = 1$. Thus, we can derive that

$$\mu^\star = \frac{\sum_{n=1}^{N} \frac{V_n}{T_n^{\max}} \sqrt{\frac{C_n}{R_n}}}{\left(1 - \tau^{\mathrm{s}} F^{\mathrm{s}} - \sum_{n=1}^{N} \frac{V_n}{T_n^{\max} R_n}\right)}, \tag{46}$$

which ends the proof.

Appendix C
Proof of Lemma 2

We aim to find an upper bound of F^{s}, which is equivalent to find the lower bound of the required communication resource for computation tasks. Therefore, we consider a special case where all devices have the highest data rate, i.e., $\min_n R_n$, and the lowest computation load, i.e.,

$\left(\min_n V_n, \min_n C_n, \max_n T_n^{\max}\right)$, and all computation resource is allocated to the devices. Then, according to (4), the devices need less communication resource in this case compared with the optimal solution. The corresponding required communication resource of computation tasks can be given by

$$\frac{N \min_n V_n f^{\mathrm{e}}}{\min_n R_n \left(\max_n T_n^{\max} f^{\mathrm{e}} - N \min_n V_n \min_n C_n\right)}. \tag{47}$$

Thus, we can express an upper bound of F^{s} as (35), which ends the proof.

References

[1] Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao, "Application of machine learning in wireless networks: Key techniques and open issues," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3072–3108, 4th Quart., 2019.

[2] G. Zhu, D. Liu, Y. Du, C. You, J. Zhang, and K. Huang, "Towards an intelligent edge: Wireless communication meets machine learning," *IEEE Commun. Mag.*, vol. 58, no. 1, pp. 19–25, Jan. 2020.

[3] D. Korzun, E. Balandina, A. Kashevnik, S. Balandin, and F. Viola, *Ambient Intelligence Services in IoT Environments: Emerging Research and Opportunities*. Hershey, PA, USA: IGI Global, Jun. 2019.

[4] Y. Ma, G. Zhou, and S. Wang, "WiFi sensing with channel state information: A survey," *ACM Comput. Surv.*, vol. 52, no. 3, pp. 1–36, May 2020.

[5] J. Liu, C. Xiao, K. Cui, J. Han, X. Xu, and K. Ren, "Behavior privacy preserving in RF sensing," *IEEE Trans. Depend. Sec. Comput.*, vol. 20, no. 1, pp. 784–796, Jan.–Feb. 2023.

[6] Y. Cui, F. Liu, X. Jing, and J. Mu, "Integrating sensing and communications for ubiquitous IoT: Applications, trends, and challenges," *IEEE Netw.*, vol. 35, no. 5, pp. 158–167, Sep. 2021.

[7] F. Liu *et al.*, "Integrated sensing and communications: Towards dual-functional wireless networks for 6G and beyond," *IEEE J. Sel. Areas Commun*, vol. 40, no. 6, pp. 1728–1767, Jun. 2022.

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

[9] 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.

[10] P. Liu, G. Zhu, S. Wang, W. Jiang, W. Luo, H. V. Poor, and S. Cui, "Toward ambient intelligence: Federated edge learning with task-oriented sensing, computation, and communication integration," *IEEE J. Sel. Topics Signal Process.*, vol. 17, no. 1, pp. 158–172, Jan. 2023.

[11] D. Wen, P. Liu, G. Zhu, Y. Shi, J. Xu, Y. C. Eldar, and S. Cui, "Task-oriented sensing, computation, and communication integration for multi-device edge AI," 2022, *arXiv:2207.00969*.

[12] W. Xu, Z. Yang, D. W. K. Ng, M. Levorato, Y. C. Eldar, and M. Debbah, "Edge learning for B5G networks with distributed signal processing: Semantic communication, edge computing, and wireless sensing," *IEEE J. Sel. Top. Signal Process.*, vol. 17, no. 1, pp. 9–39, Jan. 2023.

[13] Q. Luo, C. Li, T. H. Luan, and W. Shi, "Minimizing the delay and cost of computation offloading for vehicular edge computing," *IEEE Trans. Serv. Comput.*, vol. 15, no. 5, pp. 2897–2909, Sep. 2022.

[14] Y. Fu, C. Li, F. R. Yu, T. H. Luan, and Y. Zhang, "A survey of driving safety with sensing, vehicular communications, and artificial intelligence-based collision avoidance," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6142–6163, Jul. 2022.

[15] B. Li, A. P. Petropulu, and W. Trappe, "Optimum co-design for spectrum sharing between matrix completion based MIMO radars and a MIMO communication system," *IEEE Trans. Signal Process.*, vol. 64, no. 17, pp. 4562–4575, Sep. 2016.

[16] Y. He, Y. Cai, H. Mao and G. Yu, "RIS-assisted communication radar coexistence: Joint beamforming design and analysis," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 7, pp. 2131–2145, Jul. 2022.

[17] M. Kotaru, K. Joshi, D. Bharadia, and S. Katti, "SpotFi: Decimeter level localization using WiFi" *ACM SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, pp. 269–282, 2015.

[18] Y. Wang, K. Wu, and L. M. Ni, "WiFall: Device-free fall detection by wireless networks," *IEEE Trans. Mob. Comput.*, vol. 16, no. 2, pp. 581–594, Feb. 2017.

[19] Y. Xie, M. Li, J. Xiong, and K. Jamieson, "MD-Track: Leveraging multi-dimensionality in passive indoor Wi-Fi tracking," *Proc. Annu. Int. Conf. Mob. Comput. Netw. (MobiCom)*, Aug. 2019, pp. 1–16.

[20] F. Liu, L. Zhou, C. Masouros, A. Li, W. Luo, and A. Petropulu, "Toward dual-functional radar-communication systems: Optimal waveform design," *IEEE Trans. Signal Process.*, vol. 66, no. 16, pp. 4264–4279, Aug. 2018.

[21] F. Liu, C. Masouros, A. P. Petropulu, H. Griffiths, and L. Hanzo, "Joint radar and communication design: Applications, state-of-the-art, and the road ahead," *IEEE Trans. Commun.*, vol. 68, no. 6, pp. 3834–3862, Jun. 2020.

[22] F. Liu, C. Masouros, A. Li, H. Sun, and L. Hanzo, "MU-MIMO communications with MIMO radar: From co-existence to joint transmission," *IEEE Trans. Wireless Commun.*, vol. 17, no. 4, pp. 2755–2770, Apr. 2018.

[23] Y. He, Y. Cai, G. Yu, and K. -K. Wong, "Joint transceiver design for dual-functional full-duplex relay aided radar-communication systems," *IEEE Trans. Commun.*, vol. 70, no. 12, pp. 8355–8369, Dec. 2022.

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

[25] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3590–3605, Dec. 2016.

[26] 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.

[27] J. Zhang et al., "Joint resource allocation for latency-sensitive services over mobile edge computing networks with caching," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4283–4294, Oct. 2018.

[28] C. Ding, J. Wang, H. Zhang, M. Lin, and G. Li, "Joint MIMO precoding and computation resource allocation for dual-function radar and communication systems with mobile edge computing," *IEEE J. Select. Areas Commun.*, vol. 40, no. 7, pp. 2085–2102, Jul. 2022.

[29] N. Huang, T. Wang, Y. Wu, Q. Wu, and T. Q. S. Quek, "Integrated sensing and communication assisted mobile edge computing: An energy efficient design via intelligent reflecting surface," *IEEE Wireless Commun. Lett.*, vol. 11, no. 10, pp. 2085–2089, Oct. 2022.

[30] R. Xiao, J. Liu, J. Han, and K. Ren, "OneFi: One-shot recognition for unseen gesture via COTS Wi-Fi," in *Proc. Conf. Embed. Networked Sens. (SenSys)*, Coimbra, Portugal, Nov. 2021, pp. 206–219.

[31] X. Wang, K. Sun, T. Zhao, W. Wang, and Q. Gu, "Dynamic speed warping: Similarity-sased one-shot learning for device-free gesture signals," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Toronto, Canada, 2020, pp. 556–565.

[32] C. Feng, N. Wang, Y. Jiang, X. Zheng, K. Li, Z. Wang, and X. Chen, "Wi-Learner: Towards one-shot learning for cross-domain Wi-Fi based gesture recognition," in *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol. (IMWUT)*, vol. 6, no. 3, pp. 1–27, Sep. 2022.

[33] M. Li, Y. Meng, J. Liu, H. Zhu, X. Liang, Y. Liu, and N. Ruan, "When CSI meets public WiFi: Inferring your mobile phone password via WiFi Signals," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, Oct. 2016, pp. 1068–1079.

[34] J. Liu, Y. He, C. Xiao, J. Han, L. Cheng, and K. Ren, "Physical-world attack towards WiFi-based behavior recognition," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Honolulu, London, United Kingdom, Jun. 2022, pp. 400–409.

[35] Y. He, J. Ren, G. Yu, and Y. Cai, "D2D communications meet mobile edge computing for enhanced computation capacity in cellular networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 3, pp. 1750–1763, Mar. 2019.

[36] C. B. Barneto, L. Anttila, M. Fleischer, and M. Valkama, "OFDM radar with LTE waveform: Processing and performance," in *Proc. IEEE Radio Wireless Symp. (RWS)*, Jan. 2019, pp. 1–4.

[37] S. Tan, Y. Ren, J. Yang, and Y. Chen, "Commodity WiFi sensing in 10 years: Status, challenges, and opportunities," *IEEE Internet Things J.*, vol. 9, no. 18, pp. 17832–17843, Sep. 2022.

[38] M. Jankiraman, *FMCW Radar Design*. Norwood, MA, USA: Artech House, 2008.

[39] Y. Zheng, Y. Zhang, K. Qian, G. Zhang, Y. Liu, C. Wu, and Z. Yang, "Zero-effort cross-domain gesture recognition with Wi-Fi," in *Proc. Annu. Int. Conf. Mobile Syst. Appl. Services (MobiSys)*, New York, NY, USA, 2019, pp. 313–325.

[40] A. V. Oppenheim, A. S. Willsky, and S. H. Nawab, *Signals and Systems*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 1996.

[41] R. Seri, "A tight bound on the distance between a noncentral chi square and a normal distribution," *IEEE Commun. Lett.*, vol. 19, no. 11, pp. 1877–1880, Nov. 2015.

[42] G. Li, S. Wang, J. Li, R. Wang, F. Liu, M. Zhang, X. Peng, and T. X. Han. "Rethinking the tradeoff in integrated sensing and communication: Recognition accuracy versus communication rate," 2021, *arXiv:2107.09621.*

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

[44] Y. He, J. Ren, G. Yu, and Y. Cai, "Optimizing the learning performance in mobile augmented reality systems with CNN," *IEEE Trans. Wireless Commun.*, vol. 19, no. 8, pp. 5333–5344, Aug. 2020.

[45] *LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Channels and Modulation (3GPP TS 36.211 Version 15.6.0 Release 15)*, document TS 36 211 V15.6.0, 3GPP, Jul. 2019.

[46] J. Kiefer, "Sequential minimax search for a maximum," in *Proc. Amer. Math. Soc. (AMS)*, 1953, pp. 502–506.