# A Sustainable Multi-modal Multi-layer Emotion-aware Service at the Edge

Long Hu, Wei Li, Jun Yang, Giancarlo Fortino, Senior Member, IEEE, and Min Chen, Senior Member, IEEE

Abstract—Limited by the computational capabilities and battery energy of terminal devices and network bandwidth, emotion recognition tasks fail to achieve good interactive experience for users. The intolerable latency for users also seriously restricts the popularization of emotion recognition applications in the edge environments such as fatigue detection in auto-driving. The development of edge computing provides a more sustainable solution for this problem. Based on edge computing, this article proposes a multi-modal multi-layer emotion-aware service (MULTI-EASE) architecture that considers user's facial expression and voice as a multi-modal data source of emotion recognition, and employs the intelligent terminal, edge server and cloud as multi-layer execution environment. By analyzing the average delay of each task and the average energy consumption at the mobile device, we formulate a delay-constrained energy minimization problem and perform a task scheduling policy between multiple layers to reduce the end-to-end delay and energy consumption by using an edge-based approach, further to improve the users' emotion interactive experience and achieve energy saving in edge computing. Finally, a prototype system is also implemented to validate the architecture of MULTI-EASE, the experimental results show that MULTI-EASE is a sustainable and efficient platform for emotion analysis applications, and also provide a valuable reference for dynamic task scheduling under MULTI-EASE architecture.

Index Terms—computing offloading, edge computing, emotion recognition, energy saving, multi-modal data

## **1** INTRODUCTION

Emotion recognition is a complex process, that is aimed at the individual's emotional state, which means that the emotions corresponding to each individual's behavior are different [1]. With the rapid development of artificial intelligence, many tasked related to computer vision, speech recognition, and natural language processing have achieved great success [2]–[4], but emotional modeling for individuals still faces great challenges. Presently, the main methods for emotion recognition include speech emotion recognition [5], text emotion recognition [6], facial emotion recognition [7], gesture emotion recognition [8], etc. As the emotional data source is very rich, the single-modal model can not judge the user's emotion well, so we need multi-modal data to learn the emotion model. In this paper, the emotion analysis is based on multi-modal data such as user's facial expression and voice.

In addition, the emotion recognition task cannot achieve a good interactive experience due to the undesirable interactive latency. To solve the problem, we introduce edge computing and put forward a multi-layer model for algorithm execution, that is emotional services can be provided at device terminals, network edges and clouds [9], [10]. The emergent paradigm of edge computing [11], [12], [15] advocates that computational and storage resources can be extended to the edge of the network so that the impact of data transmission latency over the Internet can be effectively reduced for time-constrained applications, i.e., emotion analysis [13]. For example, in auto driving scene, the virtual emotional robot needs to collect the driver's voice, facial

expression and other multi-modal emotional data, as well as some surrounding environmental information, including the driver's current road condition information, geographical location, time information, etc., which has strong demands on computational and storage resources, as well as low latency. Therefore, we deploy some AI algorithms at the edge computing nodes to handle emotion recognition tasks [14]. These algorithms usually have lower performance than algorithms deployed in the cloud, but edge computing nodes can quickly feed back results to users.

However, with the widespread deployment of edge computing devices, the energy demand of these devices has increased and started to become a noticeable issues for sustainable development of time-constrained IoT applications [16]-[19]. Energy efficiency optimization for such applications becomes more challenging when considering the rapid constant grow of edge devices/sensors [20], [21]. For example, the current number of IoT devices will rapidly increase from 15 billion to 50 billion by 2020 (according to CISCO), while the number of sensors will increase to as high as 1 trillion by 2030 (according to HP Labs). In emotion analysis applications [22], we need multi-modal data for the emotion recognition, such application consumes lots of sensors for acquiring emotion-related data such as facial expression, voice and other physical data. The sustainability of such systems becomes a necessity. Although local computing consumes more energy, it can significantly minimize the execution latency without additional communication or waiting delay. Thus, it is critical to make efficient offloading decision between energy consumption of smart mobile devices and execution latency of the corresponding tasks [23]-[25].

Above all, a flexible task scheduling strategy must be adopted considering the delay and energy demands [26], [27]. The delay, energy consumption and quality of service (such as the accuracy of emotion recognition in this paper) are different at different layer levels, and the user's sensitivity to the delay of emotion recogni-

Long Hu, Wei Li, Jun Yang, and Min Chen are with School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China (longhu.cs@gmail.com, weili\_epic@hust.edu.cn, junyang\_cs@hust.edu.cn, minchen2012@hust.edu.cn).

Giancarlo Fortino is with University of Calabria (g.fortino@unical.it).

<sup>•</sup> Min Chen is the corresponding author.

tion is also different. At the same time, considering the energy consumption problem, we hope to find a kind of compromise scheduling scheme to enable energy consumption to be reduced while satisfying user delay requirements and service quality requirements. Therefore, in this paper, we propose a multi-modal multi-layer emotion-aware services (MULTI-EASE) and perform a task scheduling policy between multiple layers by formulating a delay-constrained energy minimization problem which minimizes the total energy consumption of the system subjected to the latency and emotion service quality constrains. The main contributions of this article are as follows:

- We propose a MULTI-EASE architecture and perform a task scheduling policy between multiple layers to reduce the end-to-end delay and energy consumption by using an edge-based approach;
- We formulate a delay-constrained energy minimization problem to improve the users' emotion interactive experience and achieve energy saving;
- We deploy a prototype system to validate the architecture of MULTI-EASE that is proved to be a sustainable and efficient platform for emotion analysis applications.

The remainder of this article is organized as follows. Section 2 introduces the MULTI-EASE architecture. Section 3 introduces the modeling of multi-layer task scheduling and the analysis of delay and energy. Section 4 presents the MULTI-EASE prototype. Section 5 discusses the testing of main MULTI-EASE components. Finally, Section 6 concludes the article.

## 2 MULTI-EASE ARCHITECTURE

Based on edge computing, this paper proposes a multi-modal multi-layer emotion-aware service (MULTI-EASE) architecture that considers multi-modal data as the data source of emotion recognition, and employs the intelligent terminal, edge server and cloud as multi-layer execution environment. The architecture of MULTI-EASE is shown in Fig. 1. The MULTI-EASE conducts a long-term intelligent and personalized emotion perception of a single user from multi-model data, and meanwhile performs a task scheduling policy between multiple layers to optimize the emotion recognition task. Adaptive information fusion technologies [28] can also be integrated into MULTI-EASE in order to enhance the emotion perception accuracy. Details of MULTI-EASE architecture are introduced as follows.

#### 2.1 Multi-modal

At present, the emotion data sources are very abundant, as shown in Fig. 1, the data could be audio-visual information, physiological signal, diet and sports information, etc. In this paper, the facial and audio data are adopted to achieve emotion recognition, and facial emotion detection is characterized by higher recognition accuracy and lower computation overhead, while the audio emotion detection requires high accuracy of the original audio data. This paper mainly adopts the intelligent terminal such as an interactive robot to collect audio-visual data of the user and assist in better detection and recognition of user emotion.

#### 2.2 Multi-layer

After acquiring the emotion data, a terminal always has to offload the computing tasks. Networking between different edge terminals can be implemented based on some existing advanced energyefficient communication mechanisms such as [29]. The emotion data of users are personalized, and the demands on delay are strict for users, so that the emotion computing tasks may be offloaded to different service nodes for processing, such as other terminals, edge servers and cloud platforms as shown in Fig. 1.

a) Terminal device: In the emotion recognition, an intelligent device or emotion recognition robot denotes as a local terminal device. The terminal device has the function of collecting the emotion data. For instance, a robot can collect facial expression and audio of a user, and the smart clothing and other wearable devices can collect the physiological data of a user. However, both computing and storage capability and the battery energy of terminal devices are limited. When the number of users' requests is excessive, or the complexity of a computing task is larger than available computing power, it is not suitable to perform the task directly on a terminal device. On the other hand, local devices are the closest devices to user, thus, they have low communication latency and are suitable for processing simple task with sensitive latency.

**b)** Edge server: A part of computing tasks can be processed on a network edge by introducing the edge computing to reduce network congestion. The edge computing layer is composed of many nodes possessing the computing capability. These edge nodes can be gateway, router, exchange or local server, etc. In emotion recognition, a local server near a terminal device is considered as an edge node. The computing capability of edge computing is weaker than that of cloud computing, but the communication latency of edge server is far below that of a cloud platform. The performance of algorithm deployed on the edge server is generally lower than that of algorithm deployed on the cloud, but algorithm deployed on the edge server can rapidly feed-back the result to a user. The computing capability and storage capability of an edge computing node are also limited. Therefore, more complex computation should be executed on a remote cloud platform.

c) Cloud platform: Both local computing and edge computing may relieve network congestion and reduce communication latency, while cloud computing does not process unnecessary computing tasks and can focus on high-precision computation and analysis to provide an optimal computing service for users. The infrastructure of cloud computing is based on a cloud platform that deploys a high-performance emotion recognition algorithm.

d) Scheduling engine: By using the SDN (Software Defined Network, SDN) technology, we can deploy a scheduling engine in three layers of a network framework. The scheduling engine processes real-time multimodal emotional data flow in network environment, and can execute task scheduling policy through perceiving the computing resources, environmental communication resources, and network resources (such as network type, business data flow, communication quality, and other dynamic environment parameters) of a three-layer structure. The scheduling engine has global information about multi-layer execution environment and determines the scheduling policy on line according to current task requests and network resources.

#### **3** MODELING AND ANALYSIS

#### 3.1 Task model

For each task  $Q_i$ , it can be expressed as  $Q_i = \{\omega_i, s_i, o_i, a_i\}$ .  $\omega_i$  is the computation resource for task  $Q_i$  and can be measured by the number of CPU cycles required to accomplish the task;  $s_i$ 



Fig. 1. MULTI-EASE Architecture

specifies the size of the task in bits;  $o_i$  represents the amount of data generated by the task feedback;  $a_i$  represents the quality of task performance. In emotion detection ,  $\omega_i$  is the computation resource required for the emotion detection task;  $s_i$  is the emotion data (image, voice, video, etc.);  $o_i$  is the amount of data fed back to the client; and  $a_i$  is the recognition accuracy of emotion detection. In addition,  $d_i$  represents the task duration of task  $Q_i$ . Considering that the output of a task is generally much smaller than the task size, and can be returned to the device with negligible transmission delay (e.g., face recognition and language processing), the feedback time can be represented by  $\xi_i(o_i)$  which is viewed as a constant and can be ignored.

The delay and energy consumption in different layers will be discussed below.

#### 3.2 Multi-layer task scheduling

## 3.2.1 Local Layer

The latency of emotion computing task on a terminal device is given by (1). In (1),  $T_{n,i}^{loc}$  is the time required to perform the task locally at the terminal device n. We assume that the CPU at the nth local device is operating at frequency  $f_n^{loc}$  (in Hz) if a task is being executed, and its energy consumption is given by  $P_n^{loc}$  (in W); otherwise, the local CPU is idle and consumes no energy. The number of required CPU cycles for executing a task successfully is denoted as  $\omega_i$ , which depends on the types of mobile applications as introduced in the task definition  $Q_i = \{\omega_i, s_i, o_i, a_i\}$ . Then the task duration of local processing can be represented by  $d_{n,i}^{loc}$ , as given by

$$d_{n,i}^{loc} = T_{n,i}^{loc} = \frac{\omega_i}{f_n^{loc}}.$$
(1)

The CPU power consumption is widely modeled to be a superlinear function of  $f_n^{loc}$ , as given by  $P_n^{loc} = \kappa (f_n^{loc})^{\gamma}$ .  $\kappa$  and  $\gamma$  are pre-configured model parameters depending on the chip architecture. According to [30], the energy consumption of device n for local computation, denoted by  $E_{n,i}^{loc}$ , is therefore given by

$$E_{n,i}^{loc} = P_n^{loc} \cdot T_{n,i}^{loc} = \kappa (f_n^{loc})^{\gamma - 1} \omega_i \tag{2}$$

Typically,  $\kappa = 10^{-26}$  denotes the energy coefficient that depends on the chip architecture [31], and  $2 \le \gamma \le 3$  [32].

#### 3.2.2 Edge Layer

In order to offload a computation task to the edge server, all the input data of the task should be successfully delivered to the edge server over the wireless channel. We assume  $P_n^{tr}$  is the uplink transmission power of terminal device n,  $h_{n,m}$  denotes the channel power gain between the device n and the edge server m, B is the system bandwidth and  $N_0$  is the noise power spectral density at the receiver; According to the work in [33], the uplink data rate  $r_{n,m}$  of device n can be given by

$$r_{n,m} = Blog_2(1 + \frac{P_n^{tr} h_{n,m}}{N_0})$$
(3)

The latency of edge computing is given by (4) where  $T_{n,m}^{loc \rightarrow edge}$  is the time needed to offload emotion data to a edge server m.  $T_{m,i}^{edge}$  is the execution time of task i on edge server m,  $T_{m,n}^{down}$  is the time needed to feedback the computation results to terminal device n. Assume that  $f_m^{edge}$  represents the computing capacity of the mth edge node, then the task duration on edge node can be represented by  $d_{m,i}^{edge}$ .

$$d_{i}^{edge} = T_{n,m}^{loc \to edge} + T_{m,i}^{edge} + T_{m,n}^{down} = \frac{s_{n,m}}{r_{n,m}} + \frac{\omega_{i}}{f_{m}^{edge}} + \xi_{i}(o_{i}).$$
(4)

The energy consumption can be calculated as the communication consumption between the local device and edge node. Let  $P_n^{tr}$  denotes the transmission power of terminal device nand  $P_n^{id}$  denotes the idle power. When the task is executed on the edge server, the mobile device needs to wait for the return of the response result. The idle power consumption of the mobile device can be calculated as  $P_n^{id}T_{m,i}^{edge}$ . Then, the energy consumption is defined as the sum of idle power consumption and data transmission power consumption which can be expressed as  $E_{m,i}^{edge}$  as follows:

$$E_{m,i}^{edge} = P_n^{tr} T_{n,m}^{loc \to edge} + P_n^{id} T_{m,i}^{edge} = P_n^{tr} \frac{s_{n,m}}{r_{n,m}} + P_n^{id} \frac{\omega_i}{f_m^{edge}}$$
(5)

#### 3.2.3 Cloud Layer

The latency of cloud computing is given by (6), where  $T_{n,k}^{loc \rightarrow cloud}$  is the time for emotion data offloading to the remote cloud server k, and  $T_{k,i}^{cloud}$  is the execution time of task i on cloud server k. Assume that  $f_k^{cloud}$  represents the computing capacity of the kth cloud server, then the task duration on cloud can be represented by  $d_{k,i}^{cloud}$ .

$$d_{k,i}^{cloud} = T_{n,k}^{loc \to cloud} + T_{k,i}^{cloud} + T_{k,n}^{down} = \frac{s_{n,k}}{r_{n,k}} + \frac{\omega_i}{f_k^{cloud}} + \xi_i(o_i).$$
(6)

The energy consumption can be calculated as the communication consumption between the local device and cloud, as the same case of edge computing, the energy consumption is defined by  $E_{k,i}^{cloud}$  as follows:

$$E_{k,i}^{cloud} = P_n^{tr} T_{n,k}^{loc \to cloud} + P_n^{id} T_{k,i}^{cloud} = P_n^{tr} \frac{s_{n,k}}{r_{n,k}} + P_n^{id} \frac{\omega_i}{f_k^{cloud}}$$
(7)

#### 3.3 Delay and Power Analysis

In this subsection, we will analyze the average delay of each task and the average power consumption at the mobile device by modeling the MULTI-EASE system.

Let  $p_i^{loc}, p_i^{edge}, p_i^{cloud}$  represent the offloading policy of task  $Q_i$ . If  $Q_i$  is processed by a local device n, then  $p_{n,i}^{loc}$  equals to 1, otherwise it equals to 0. Similarly, if  $Q_i$  is processed by a edge m, then  $p_{m,i}^{edge}$  equals to 1, otherwise it equals to 0. If  $Q_i$  is processed by a cloud k then  $p_{k,i}^{cloud}$  is 1, otherwise it equals to 0. Also, the sum of  $p_i^{loc}p_i^{edge}p_i^{cloud}$  should equal to 1 as defined by (11).

$$p_{n,i}^{loc} = \begin{cases} 1, & \text{if task } i \text{ is processed by local device } n, \\ 0, & \text{otherwise.} \end{cases}$$
(8)

$$p_{m,i}^{edge} = \begin{cases} 1, & \text{if task } i \text{ is processed by edge } m, \\ 0, & \text{otherwise.} \end{cases}$$
(9)

$$p_{k,i}^{cloud} = \begin{cases} 1, & \text{if task } i \text{ is processed by cloud } k, \\ 0, & \text{otherwise.} \end{cases}$$
(10)

$$p_{n,i}^{loc} + p_{m,i}^{edge} + p_{k,i}^{cloud} = 1.$$
 (11)

Then, the task duration  $d_i$  and energy consumption  $E_i$  of task  $Q_i$  can be expressed by:

$$d_{i} = p_{n,i}^{loc} d_{n,i}^{loc} + p_{m,i}^{edge} d_{m,i}^{edge} + p_{k,i}^{cloud} d_{k,i}^{cloud}.$$
 (12)

$$E_{i} = p_{n,i}^{loc} E_{n,i}^{loc} + p_{m,i}^{edge} E_{m,i}^{edge} + p_{k,i}^{cloud} E_{k,i}^{cloud}.$$
 (13)

Considering the delay, energy consumption and quality of service are different at different layer levels, and the user's sensitivity to the delay of emotion recognition is also different. We formulate a delay-constrained energy minimization problem to minimize the energy consumption while satisfying user delay requirements and accuracy of algorithm. Let assume user's requirement be defined as  $U_i = \{A_i, D_i\}$ , where  $A_i$  represents the requirement for task performance (accuracy in emotion detection), and  $D_i$  is the task deadline, i.e., the maximum delay that the task can tolerate.

$$d_i \le D_i. \tag{14}$$

$$a_i \ge A_i. \tag{15}$$

Assume that the total number of tasks initiated within the service area is q. An energy efficiency optimization problem, which is to minimize the overall energy consumption while satisfying user's requirement, can be formulated by:

minimize: 
$$\frac{1}{q} \sum_{i=1}^{q} E_i$$
, (16)  
subject to: (2) - (15).

The problem defined by (16), is a typical 0-1 integer linear optimization problem. By the use of branch and bound algorithm [34], the optimal solution can be obtained. Thus, the linear iterative algorithm can be utilized and obtain the approximate optimal solution.

#### 4 MULTI-EASE TEST PLATFORM

#### 4.1 Prototype system

The MULTI-EASE prototype platform is shown in Fig. 2, where it can be seen that intelligent terminal and local server are two typical edge computing nodes, cloud platform is a data center and GPU server is an analysis server.

The MULTI-EASE hardware parameters presented in Table 1 show the hardware profile of MULTI-EASE prototype platform and list the computing performance of a device under three different levels (terminal device, local server (edge node), and cloud platform). Each MULTI-EASE layer has different computing capability and different working load distributed to it. While generating the working load, different numbers of service requests are generated following the uniform probability distribution, and different load scenes are simulated. "Experimental parameters settings" in Table 1 shows the statistics of test executed on devices at each level. We deploy facial expression recognition and voice emotion recognition algorithms on each layer. The concurrent request time is set from 1 to 60 at Robot for facial recognition, and from 1 to 35 at Robot for voice recognition, that is because voice recognition task consumes more computation resources. Cloud has higher load capacity than local server so the concurrent invocation cloud be from 50 to 700 while it is from 50 to 550 at local server. In the experiment, we conduct several stress tests on multi-layers based on above settings.

#### 4.2 Emotion Analysis

The emotion recognition algorithms used in this paper are facial and voice emotion recognition. The model training was executed based on an independent GPU algorithm server. At the end of the training, the trained model was saved in the binary file in the form of ".pb" file. The model could be executed on a cloud



Fig. 2. MULTI-EASE Prototype

MULTI-EASE Hardware Parameters				
Node	Core Processor		Memory	
Robot	4 x ARM Cortex-A33, 1.2GHz		1GB LPDDR2	
Edge Server	Intel Cor 2 Quad 9400, 2.66GHz		8GB DIMM	
Cloud	AMD FX 8-Core, 4GHz		32 GB RAM DDR3	
Experimental Parameters Settings				
	Robot	Edge Server	Cloud	
Algorithm	Facial + Voice	Facial + Voice	Facial + Voice	
Concurrent Invocation(Facial)	1 to 60	50 to 550	50 to 700	
Concurrent Invocation(Voice)	1 to 35	50 to 550	50 to 700	

TABLE 1 MULTI-EASE Prototype Platform Profile

platform and migrated to the terminal device or other devices for execution. The model execution on the edge device needed a TensorFlow environment, and the model execution on the terminal device needed a TensorFlow support package. Figure. 3 shows the execution results of real-time emotion recognition in the terminal device. Two figures on the left side of Fig. 3 show the probability distribution of results of image emotion recognition (upper figure) and voice emotion recognition (lower figure). On the interface bottom, seven specific emotions (angry, disgust, fear, happy, sad, surprise, and neutral) are shown. The emotion mark colored in red is currently recognized emotion. The image used in the test was captured by a self-contained camera of a terminal device. The OpenCV can catch the human face in the image, conduct the corresponding preprocessing, and transmit obtained result to the model as an input data. The audio data were collected by a microphone (MIC), and the collected voice data denoted an input to the emotion model for real-time recognition. The specific algorithms are introduced as below.

- Facial emotion recognition: The facial emotion recognition firstly realizes a facial expression captured by OpenCV, then realizes a face alignment by the MTCNN (Multi-task Cascaded Convolutional Networks) [35], and finally realizes the facial feature training by the VGGNET to generate a 256-dimension facial feature vector for executing the task of expression classification.
- Voice emotion recognition: First, the speech framing is conducted, then the MFCC is adopted for feature extraction, the static speech feature vector is generated, the first and second derivatives of a generated static speech feature

are evaluated, and a three-channel speech spectrogram is obtained. Finally, the AlexNet DCNN (deep convolutional neural network) model for emotion feature extraction is adopted to generate 512-dimension emotion feature vector for executing the task of speech emotion classification [36].

## 5 PERFORMANCE EVALUATION OF EMOTION RECOGNITION

We introduce three baseline task scheduling policies, including the **local execution policy**, which executes all the computation tasks locally at the mobile device; the **cloud execution policy**, where all the tasks are offloaded to the cloud for computing; and **random execution policy**, which randomly select one layer for task execution. Our proposed task scheduling policy called **edgebased multi-layer execution policy** solves the delay-constrained energy minimization problem to reduce the end-to-end delay and energy consumption towards specific latency and energy demands of emotion analysis task.

For large scale performance analysis and evaluation, the numerical simulations are designed except for the real prototype system. The simulations are deployed based on real-world settings according to our real prototype system. All the parameters, including the energy consumption rates and computing capacity, are measured from real mobile devices. In the experimental setup, we assume that system bandwidth B is 1 MHz, the local processing power  $P_n^{loc}$ , the standby power  $P_n^{id}$ , and the transmitting power  $P_n^{tr}$  of mobile device are 0.9 W, 0.3 W and 1.3 W respectively. The



Fig. 3. Emotion Recognition Demo

corresponding Gaussian channel noise  $N_0$  and channel power gain  $h_{n,m}$  are  $10^{-9}$  W and  $10^{-5}$ , respectively. For task  $Q_i$ , we assume that required computing capacity  $w_i$  and data size  $s_i$  are generated by a probability distribution [37]. Furthermore, we assume that the computing capabilities of cloud server  $f_n^{cloud}$ , edge server  $f_n^{edge}$  and mobile device  $f_n^{loc}$  are 15GHz, 10 GHz and 2 GHz, respectively.

Below we discuss the task computing time at different layers, and the analysis of task delay and energy consumption. For convenience, we illustrate the performance indexes in detail in Table 2.

#### 5.1 Stress testing for task computing time

The performance evaluation of emotion recognition algorithm using the MULTI-EASE system is based on facial emotion and voice emotion analysis. We deploy facial expression recognition and voice emotion recognition algorithms on each layer. The stress testing for task computing time at different layers can be seen from Fig. 4.

It is obvious that the local computing time increases sharply when the concurrent request number increases to 60 for facial recognition, and to 35 for voice recognition, that is due to the limited computational capabilities of local device, and voice recognition task consumes more computation resources which results in that it cannot response much concurrent requests as facial recognition task. When the concurrent request number increases to 550 and 700 respectively, the edge computing time and cloud computing time increase sharply. Cloud has higher load capacity than local server so the concurrent invocation cloud be from 50 to 700 while it is from 50 to 550 at local server. It can also be seen that voice recognition task needs much more computing time for its higher computation demands.

## 5.2 Delay and energy analysis with task scale

The average delay is shown in Fig. 5(a), it can be observed from the figure that, the average delays achieved by the local execution, cloud execution, random execution and the edge-based multilayer execution, increase with the average computation task arrival number q, which is in accordance with our intuition. The task delay of local execution policy is much larger than that of other



(a) Task computing time for facial emotion recognition



(b) Task computing time for voice emotion recognition

Fig. 4. Stress testing for task computing time at different Layers

TABLE 2 Performance Indexes Illustration

Indexes	Symbols	Illustration	
	$T^{loc}$	Task computing time at local layer	
Task computing time	$T^{edge}$	Task computing time at edge layer	
	$T^{cloud}$	Task computing time at cloud layer	
Task delay	$\sum_{i=1}^{q} d_i$	$d_i$ is the task duration of task $Q_i$ defined in (12)	
Task energy consumption	$\sum_{i=1}^{q} E_i$	$E_i$ is the task energy consumption of task $Q_i$ defined in (13)	



(a) Task delay with task arrival number q

(b) Energy consumption with task arrival number q, where s=10 MB



(c) Energy consumption with task arrival number q, where s=110 MB

Fig. 5. Delay and energy analysis with task scale

policies, this is due to the fact that the execution time required by cloud server and edge server is much smaller than that by the local CPU. When q increases, more arrived tasks should be sent to the edge server or cloud server for lower latency.

In the energy efficiency performance aspects, simulations results show the superiority of the proposed scheme. Fig. 5(b)and Fig. 5(c) show the energy consumption of the mobile device when the number of computation task increases from 50 to 500 with four different task offloading policies, considering two cases with different task data size, i.e.,  $s_i = 10MB$  and  $s_i = 110MB$ . It is obvious that the energy consumption of our proposed policy is the lowest in both two cases compared with other three policies. However, the gap of energy consumption between local execution policy and cloud execution policy is different. It can be seen that the energy consumption of local execution policy is much larger than that of cloud execution policy when the task data size is small, while it is almost the same and even lower than that of cloud execution policy when the task data size is large. This is because it results in large transmission latency and energy when the task data size is too large.

#### 5.3 Performance analysis on energy consumption

In this subsection, we will discuss the influence of different factors on energy consumption, i.e., the task deadline, the task data size and the task computation amount. Obviously, the proposed method can always find a better energy-saving solution than other three approaches with the changes of different factors as shown in Fig. 6.

Fig. 6(a) shows the average energy consumptions when task deadline D ranges from 1s to 3.5s. With the growth of D, the energy consumptions of edge-based multi-layer execution policy first decrease and then stabilize at 0.95 J when  $D \ge 1.6s$ . This is because increasingly relaxed deadlines allow a growing number of devices to be selected for offloading, thereby reducing energy consumption. When the deadlines are so loose that almost all tasks can be selected for offloading, it can be seen that our policy has lower energy consumption compared with cloud execution policy when deadlines are loose.

Fig. 6(b) shows the average energy consumptions when task data size s ranges from 10 to 80 MB. With the growth of s, the energy consumptions of edge-based multi-layer execution policy first increase along with that of cloud execution policy, and then stabilize at 2.3 J when  $s \ge 55MB$ . It indicates that in our proposed scheduling policy, the task is offloaded to edge server or cloud server when the task data size is small and is executed locally when the data size is larger than 55 MB.

Fig. 6(c) shows the average energy consumptions when task computation amount s ranges from 0.1 to 10 gigacycles. With the growth of w, the energy consumptions of edge-based multi-layer execution policy first increase along with that of local execution policy, and then slowly increase when  $w \ge 5gigacycles$ . When the task computation amount is small, the task should better be processed locally for lower energy consumption caused by data transmission. In contrast, when  $w \ge 5gigacycles$ , the energy consumption of cloud execution is smaller due to its large computational capabilities resulting in lower latency and lower idle energy consumption for terminal device. The average energy consumption achieved by the random execution policy fluctuates in all cases since it randomly selects one layer for task execution.

## 6 CONCLUSION

In this article, the MULTI-EASE architecture that could reduce the end-to-end delay and energy consumption by using an edge-based approach is presented. By analyzing the average delay of each task and the average power consumption at the mobile device, we formulated a delay-constrained energy minimization problem which minimizes the total energy consumption of the system subjected to the latency and emotion service quality constrains. The multi-layer solution can find a kind of compromise scheduling scheme in terms of processing power availability than singlelayer at device, edge server and cloud, by considering that the delay, energy consumption and quality of service are different at different layer levels, and the user's sensitivity to the delay of emotion recognition is also different. A prototype system is also implemented to validate the architecture of MULTI-EASE to be a sustainable and efficient platform for emotion analysis applications by comparing it with systems not using such edgebased scheme. It is shown that the multi-layer solution always find a better energy-saving solution than other three approaches like local execution policy, cloud execution policy and random execution policy.

However, in this paper, we did not consider the storage capability of multi-layer execution environment, because for the emotional recognition application, the recognition model is trained offline, only real-time emotional data are needed for processing. So for small scale concurrent requests, different layers are able to store the data needed to be processed. But when the request numbers are very huge and high amount of data need to be processed, we still need to consider the case, which would be our future work that will consider the storage capability.

## REFERENCES

- M. Chen, J. Zhou, G. Tao, et al., "Wearable Affective Robot," *IEEE Access*, Vol. 6, pp. 64766–64776, 2018.
- [2] A. Krizhevsky, I. Sutskever, G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Advances in Neural Information Processing Systems 25*, pp. 1097–1105, 2012.
- [3] D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, et al., "Deep Speech 2: End-to-End Speech Recognition in English and Mandarin," *Proceedings of The 33rd International Conference on Machine Learning*, Vol. 48, pp. 173–182, 2016.
- [4] F. Meng, Z. Lu, M. Wang, et al., "Encoding source language with convalutional neural network for machine translation," arXiv preprint arXiv:1503.01838, 2015.
- [5] Z. Huang, M. Dong, Q. Mao, et al., "Speech emotion recogniton using CNN," *Proceedings of the ACM International Conference on Multimedia*, ACM, 2014.
- [6] X. Li, J. Pang, B. Mo, and Y. Rao, "Hybrid neural networks for social emotion detection over short text," *Proc. Int. Joint Conf. Neural Netw*, pp. 537–544, 2016.
- [7] Y. Huang, H. Lu, "Deep learning driven hypergraph representation for image-based emotion recognition," *Proceedings of the International Conference on Multimodal Interaction*, pp. 467-474, 2015.
- [8] D. Wu, L. Pigou, P.J. Kindermans, et al., "Deep dynamic neural networks for multimodal gesture segmentation and recognition," *IEEE Transactions* on Pattern Analysis and Machine Intelligence, Vol. 38, No. 8, pp. 1583– 1597, 2016.
- [9] Y. Zhang, et al., "TempoRec: Temporal-Topic Based Recommender for Social Network Services," *Mobile Networks and Applications*, Vol. 22, No. 6, pp. 1182-1191, 2017.
- [10] R. Casadei, G. Fortino, D. Pianini, W. Russo, C. Savaglio, M. Viroli, "Modelling and simulation of Opportunistic IoT Services with Aggregate Computing," *Future Generation Comp. Syst*, Vol 91, pp 252-262 ,2019
- [11] W. Shi, et al., "Edge Computing: Vision and challenges," *IEEE Internet Things J.*, Vol. 3, No. 5, pp. 637C-646, 2016.
- [12] M. Chen, W. Li, G. Fortino, et al., "A Dynamic Service-Migration Mechanism in Edge Cognitive Computing," ACM Transactions on Internet Technology, https://arxiv.org/pdf/1808.07198, 2018.



(a) Average energy consumption as D req increases from 1s to 3.5s.

(b) Average energy consumption as s increases from 10 to 80 MB.



(c) Average energy consumption as w increases from 0.1 to 10 gigacycles.

- Fig. 6. Performance analysis on energy consumption with different influence factors.
- [13] M. Chen, P. Zhou, G. Fortino, "Emotion Communication System," *IEEE Access* Vol. 5, pp. 326-337, 2017.
- [14] M. Chen, Y. Hao, K. Lin, et al., "Label-less Learning for Traffic Control in an Edge Network," *IEEE Network*, Vol. 32, No. 6, pp. 8–14, 2018.
- [15] P. Pace, G. Aloi, R. Gravina, G. Caliciuri, G. Fortino, A. Liotta, "An Edge-based Architecture to Support Efficient Applications for Healthcare Industry 4.0", *IEEE Transactions on Industrial Informatics*, Article in Press, 2018. DOI: 10.1109/TII.2018.2843169
- [16] M. Chen, Y. Hao, L. Hu, M. Hossain, A. Ghoneim, "Edge-CoCaCo: Towards Joint Optimization of Computation, Caching and Communication on Edge Cloud", *IEEE Wireless Communications*, Vol. 25, No. 3, pp. 21– 27, 2018.
- [17] Y. Zhang, et al., "SOVCAN: Safety-Oriented Vehicular Controller Area Network," *IEEE Communications*, Vol. 55, No. 8, pp. 94–99, 2017.
- [18] M. Hu, W. Liu, J. Lu, et al., "On the Joint Design of Routing and Scheduling for Vehicle-Assisted Multi-UAV Inspection," *Future Generation Computer Systems*, Vol.94, pp.214–223, 2019.
- [19] G. Fortino, W. Russo, C. Savaglio, W. Shen, M.C. Zhou, "Agent-Oriented Cooperative Smart Objects: From IoT System Design to Implementation," *IEEE Trans. Systems, Man, and Cybernetics: Systems*, Vol. 48 No. 11, pp. 1939-1956, 2018.
- [20] Y. Hao, Min Chen, L. Hu, M. Hossain, A. Ghoneim, "Energy Efficient Task Caching and Offloading for Mobile Edge Computing," *IEEE Access*, Vol. 6, No. 1, pp. 11365–11373, 2018.
- [21] M. Hu, Z. Chen, K. Peng, et al., "Periodic Charging for Wireless Sensor Networks with Multiple Portable Chargers," *IEEE Access*, 10.1109/AC-

CESS.2018.2885949, 2018.

- [22] M.M. Hassan, M.G.R. Alam, M.Z. Uddin, S. Huda, A. Almogren, G. Fortino, "Human emotion recognition using deep belief network architecture", *Information Fusion*, Vol. 51, pp. 10-18, 2019. DOI: 10.1016/j.inffus.2018.10.009
- [23] Y. Zhang, "Grorec: A group-centric intelligent recommender system integrating social, mobile and big data technologies," *IEEE Transactions* on Services Computing, Vol. 9, No. 5, pp. 786–795, 2016.
- [24] M. Chen, Y. Hao, "Task Offloading for Mobile Edge Computing in Software Defined Ultra-dense Network," *IEEE Journal on Selected Areas* in Communications, Vol. 36, No. 3, pp. 587–597, 2018.
- [25] Md. Golam Rabiul Alam, M. M. Hassan, Md. Zia Uddin, A. Almogren, G. Fortino, "Autonomic computation offloading in mobile edge for IoT applications," *Future Generation Comp. Syst.* Vol. 90, PP. 149-157, 2019.
- [26] K. Peng, M. Hu, C. Cai, et al., "On Simultaneous Power Replenishment for Wireless Sensor Networks With Multiple Portable Chargers," *IEEE Access*, Vol.6, pp.63120–63130, 2018.
- [27] M. Hu, W. Liu, K. Peng, et al., "Joint Routing and Scheduling for Vehicle-Assisted Multi-Drone Surveillance," *IEEE Internet of Things Journal*, 10.1109/JIOT.2018.2878602, 2018.
- [28] D. Tian, J. Zhou, Z. Sheng, "An adaptive fusion strategy for distributed information estimation over cooperative multi-agent networks," *IEEE Transactions on Information Theory*, Vol. 63, No. 5, pp. 3076–3091, 2017.
- [29] D. Tian, J. Zhou, Z. Sheng, V. C. M. Leung, "Robust energy-efficient mimo transmission for cognitive vehicular networks," *IEEE Transactions* on Vehicular Technology, Vol. 65, No. 6, pp. 3845–3859, 2016.

- [30] Y. Wen, W. Zhang, H. Luo, "Energy-optimal mobile application execution: Taming resource-poor mobile devices with cloud clones," in *Proc. IEEE INFOCOM*, pp. 2716–2720, 2012.
- [31] W. Zhang, Y. Wen, K. Guan, et al., "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Transactions on Wireless Communications*, Vol. 12, No. 9, pp. 4569–4581, 2013.
- [32] X. Chen, L. Jiao, W. Li, et al., "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, Vol. 24, No. 5, pp. 2795–2808, 2016.
- [33] B. Chen, C. Yang, "Energy Costs for Traffic Offloading by Cache-enabled D2D Communications," in : 2016 IEEE Wireless Communications and Networking Conference, 2016.
- [34] S. Boyd, L. Vandenberghe, "Convex optimization", *Cambridge university press*, 2014.
- [35] E. M. Hand, R. Chellappa, "Attributes for improved attributes: A multitask network utilizing implicit and explicit relationships for facial attribute classification," AAAI, 2017.
- [36] S Zhang, S Zhang, T Huang, et al., "Speech Emotion Recognition Using Deep Convolutional Neural Network and Discriminant Temporal Pyramid Matching," *IEEE Transactions on Multimedia*, Vol.20, No. 6, pp. 1576– 1590, 2017.
- [37] L. Tong, Y. Li, W. Gao, "A Hierarchical Edge Cloud Architecture for Mobile Computing," *IEEE INFOCOM*, pp. 399–400, 2016.



**Giancarlo Fortino** (SM12) is Full Professor of Computer Engineering at the Dept. of Informatics, Modeling, Electronics, and Systems of the University of Calabria (Unical), Italy. He received a Ph.D. in Computer Engineering from Unical, in 1995 and 2000, respectively. He is also guest professor at Wuhan University of Technology (Wuhan, China), high-end expert at HUST (China), and senior research fellow at the Italian National Research Council ICAR Institute. He is the director of the SPEME lab at Unical as

well as co-chair of Joint labs on IoT established between Unical and WUT and SMU Chinese universities, respectively. His research interests include agent-based computing, wireless (body) sensor networks, and Internet of Things. He is author of over 400 papers in intl journals, conferences and books. He is (founding) series editor of IEEE Press Book Series on Human-Machine Systems and EiC of Springer Internet of Things series and AE of many int'l journals such as IEEE TAC, IEEE THMS, IEEE IoTJ, IEEE SJ, IEEE SMCM, Information Fusion, JNCA, EAAI, etc. He is cofounder and CEO of SenSysCal S.r.I., a Unical spinoff focused on innovative IoT systems. Fortino is currently member of the IEEE SMCS BoG and of the IEEE Press BoG, and chair of the IEEE SMCS Italian Chapter.



Long Hu is a Ph.D student in School of Computer Science and Technology at Huazhong University of Science and Technology (HUST). He has also received his Master and B.S. degree in HUST. He is the Publication Chair for 4th International Conference on Cloud Computing (CloudComp 2013). Currently, his research includes 5G Mobile Communication System, Big Data Mining, Marine-Ship Communication, Internet of Things, and Multimedia Transmission over Wireless Network, etc.



**Min Chen** (M08CSM09) was an Assistant Professor with the School of Computer Science and Engineering, Seoul National University (SNU), from 2009 to 2012. He was a Post-Doctoral Fellow with SNU. He was a PostDoctoral Fellow with the Department of Electrical and Computer Engineering, University of British Columbia. He is currently a Professor with the School of Computer Science and Technology, Huazhong University of Science and Technology (HUST). He is also the Director of the Embedded and Per-

vasive Computing Laboratory. He has authored over 260 paper publica-

tions, including over 120 SCI papers, over 50 IEEE Transactions/Journal papers, six ISI highly cited papers and one hot paper. He has authored

a book on IoT, OPNET IoT Simulation (HUST Press, 2015), a book on 5G, Software Defined 5G Networks (HUST Press, 2016) and a

book on big data, Big Data Related Technologies (2014) with Springer Series in Computer Science. His Google Scholars Citations reached

over 8200 with an h-index of 45. His top paper was cited over 900

times. His research focuses on Internet of Things, mobile cloud, body area networks, emotion-aware computing, healthcare big data, cyber physical systems, and robotics. He received the Best Paper Award from the IEEE ICC 2012 and the Best Paper Runner-up Award from QShine 2008. He is currently a Guest Editor of the IEEE NETWORK and the IEEE Wireless Communications Magazine. He is the Co-Chair of the IEEE ICC 2012-Communications Theory Symposium and the Co-Chair of the IEEE ICC 2013-Wireless Networks Symposium. He is the General Co-Chair of the 12th IEEE International Conference on Computer and

Information Technology and the Mobimedia 2015. He is the General Vice

Chair of the Tridentcom 2014. He is a Keynote Speaker of CyberC 2012,

Mobiguitous 2012, and Cloudcomp 2015.



Wei Li graduated from Wuhan University of Technology (WHUT) in 2015. Now, she is a Ph.D. candidate in Embedded and Pervasive Computing Lab of Huazhong University of Science and Technology (HUST), China. Her further research includes Pervasive Computing, The Internet of things, edge computing, cognitive computing, etc.



Jun Yang received Banchelor and Master degree in Software Engineering from HUST, China in 2008 and 2011, respectively. Currently, he is a Ph.D candidate at Embedded and Pervasive Computing (EPIC) Lab in School of Computer Science and Technology, HUST. His research interests include cognitive computing, software intelligence, Internet of Things, cloud computing and big data analytics, etc.