Human in the Loop: Distributed Deep Model for Mobile Crowdsensing

Human in the Loop: Distributed Deep Model for Mobile Crowdsensing

Liangzhi Li, *Student Member, IEEE,* Kaoru Ota, *Member, IEEE,* Mianxiong Dong, *Member, IEEE*

***Abstract*—With the proliferation of mobile devices, crowdsensing has become an appealing technique to collect and process big data. Meanwhile, the rise of 5th generation wireless systems (5G), especially the new cellular base stations with computing ability, brings about the revolutionary edge computing. Although many approaches regarding the mobile crowdsensing have emerged in the last few years, very few of them are focused on the combination of edge computing and crowdsensing. In the paper, we adopt the state-of-the-art edge computing method to solve the crowdsensing problem with the real-time sensing data, and more importantly, make human be in the loop again, in order to respect the users' willing and privacy. A distributed deep learning model is adopted to extract features from the captured data, which is not only a compression process to reduce the communication cost, but an encryption procedure for safety protection. The proposed model enables the crowdsensing system to fully harness the computing capacity of edge nodes and devices, and obtain a strong data analysis ability to process the captured data. Simulations demonstrate that our approach is robust and efficient, and outperforms other strategies in several related tasks.**

***Index Terms*—Edge computing, crowdsensing, human-driven, deep learning, big data.**

I. Introduction

With the rapid development of Internet of Things (IoT), many applications have emerged to process the real-time sensing data, analyze its contents, find the hidden patterns, and ultimately, give the right labels or predict the future trends. In the other hand, crowdsensing is an economic and efficient approach to collect data on an extensive scale, and can be used as a scalable and stable method for some costly and complicated tasks in the IoT research. It has become a hot topic in recent years. The progressive development of miniaturized sensing and computing devices, especially the explosive growth of mobile phones, tablets, and wearables, gives significant prominence to the crowdsensing [1]. Both the academia and the industry have recognized its values, for example, to conduct the data collection with no large-scale investment. Many companies have utilized crowdsensing to acquire data at relatively low cost, in order to support their online service based on the captured data, some of the most notable examples being Facebook, Google and Uber. However, the boom of crowdsensing also brings about two grave problems.

Liangzhi Li, Kaoru Ota and Mianxiong Dong are with the Department of Information and Electronic Engineering, Muroran Institute of Technology, Japan.
E-mail:{16096502, ota, mxdong}@mmm.muroran-it.ac.jp
Manuscript received xx xx, 20xx; revised xx xx, 20xx.

First, the crowdsensing applications create a huge number of data, which poses significant communication and computational costs for the existing cloud infrastructure [2]. Compared with the centralized servers, the user-end devices have limited batteries and computing abilities. Therefore, existing crowdsensing systems usually put the heavy computation tasks into the centralized servers, such as the data processing and analysis. As a typical scenario in current cloud applications, the major part of the computational burden is usually shifted to the centralized servers, resulting in the rapid increase of communication frequency and server calculation load. The former brings significant traffic to the cloud infrastructure; and the latter one leads to overwhelming, sometimes unbearable, load to the centralized servers.

The other concern is about the respect of crowdsensing contributors. In fact, it has been a long time since the users are excluded in the crowdsensing process. Although most service guarantees the right to know and to decide, it is very difficult for users to truly get involved in the processing loop, e.g., to what extent the privacy should be protected, to what amount the device power can be consumed, etc. The compromise between data uploading, which may cause the privacy leaks [3], and local computing, which will result in the energy consumption of the mobile devices, is merely decided by the service providers, rather than the actual device users.

To address these problems, we adopt the state-of-the-art edge computing and deep learning methods to balance the workloads in the cloud, and give the control of crowdsensing process back to the users. Edge computing pushes calculation tasks away from the central points to the logical boundaries of the cloud; and deep learning is a good choice to conduct the data processing task, simultaneously considering the user privacy and communication cost. Therefore in the paper, we design an edge computing based deep learning system for universal crowdsensing tasks, as shown in Fig. 1. The right part is the network architecture, in which three edge nodes are connected to the central cloud, i.e., the cellular base station, the wired router and the gateway in the buildings. Through these network nodes, various edge devices can be connected to the cloud, e.g., mobile phones, sensors, smart meters, electrical appliances, etc., which is very common in the current IoT age. The difference between our method and existing crowdsensing approaches is that we successfully adapt the state-of-the-art deep learning model to the edge computing framework. As a result, with the proposed deep model, the captured data is directly processed in the edge devices and edge nodes, as is shown in the left part of Fig. 1. And only the irreversibly extracted features are uploaded to the centralized servers.

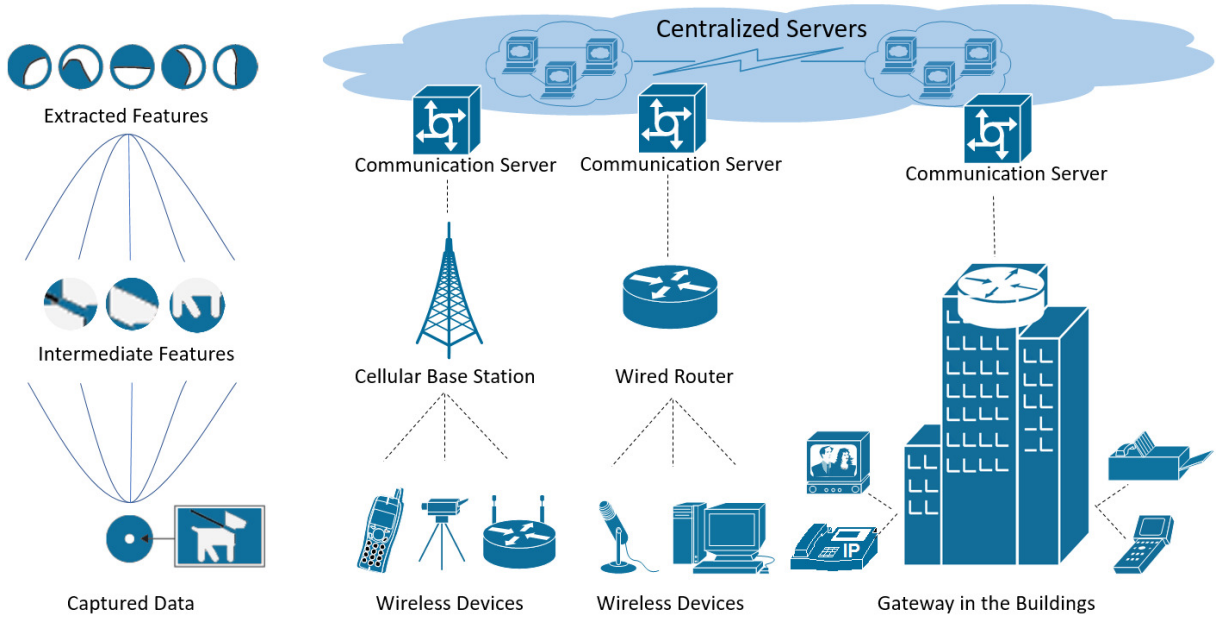


Fig. 1. The proposed edge computing based crowdsensing system. The deep model is adopted to extract features from the raw captured data for cost saving and privacy protection.

Although the centralized servers in the cloud end have strong computing abilities, we attempt to push the computing task to the user end. On the one hand, this scheme can keep the sensitive data in the user side and prevent the unauthorized access to the user privacy; on the other hand, it can also reduce the huge computing load of the centralized servers and fully utilize the computing resource in the cloud edge.

The main contributions of our work include:

- We propose a distributed deep model for the mobile crowdsensing problem. The proposed model can fully utilize the edge computing resource, and reduce the calculation load of the cloud servers.
- We transform the crowdsensing task into a hierarchical computing problem, and allocate different layers to different computing nodes. With the proposed method, crowdsensing contributors obtain the right to decide the balance between the privacy protection and energy saving.
- We work out a dynamic learning model for the changing sensing tasks. In our model, the higher layers can be flexibly modified in the centralized servers, while keeping fixed lower layers in the edge nodes and devices.

The rest of the paper is organized as follows. We first give a brief introduction regarding the existing research in the related area. After that, we present an overview of the proposed system, and then detail the human-driven strategy for mobile crowdsensing. In the experiment section, several simulations are conducted to demonstrate the performance of the proposed method. Finally, conclusions are drawn in the last section.

II. Related Works

A. Current Crowdsensing Approaches

Crowdsensing is a technology and a trend to utilize the rapidly-developing sensing and computing ability of mobile devices, such as smart phones, tablets, etc., to gather, process, calculate, and analysis the data generated by the environment, society, and other sources. Crowdsensing is usually for the public affairs or some wide-range commercial applications.

The first problem in crowdsensing is how to persuade users to participate in the projects, and encourage them to make contributions positively on data collection and sharing. Several incentive mechanisms are proposed for this purpose [4], [5]. The prime principle is to design a trading model, in which all captured data has a value and can be sold to some companies. There are two mainstream mechanisms, i.e., auction and lottery. The former one is the most common mechanism in crowdsensing. The bidders are the users who have the mobile sensing devices, and the auctioneers attempt to buy the data from the bidders. The latter one focus on the even distribution of the winning positions, and the winner is decided by a probability.

Another important concern is the sensing cost. There have been some researchers working in this area to reduce the overall sensing cost while ensuring data quality. A novel way is to leverage the temporal and spatial relationship between the data captured in different areas, in order to decrease the essential number of allocated sensing tasks [6]. A prediction framework is proposed to predict the data of unsensed area with the captured data in some selected areas.

Communication cost is also a consideration. The rapidly growing crowdsensing applications impose heavy burdens on the existing network. Therefore, the performance of these applications may deteriorate in some scenarios because of the overwhelming communication requests. A congestion-ware paradigm is worked out for dense crowdsensing to ensure load balancing and reliable communication among mobile devices [7].

TABLE I
FEATURES COMPARISON OF CROWDSENSING METHODS.

	Existing Approaches	Proposed Approach
Load	Concentrated in servers	Distributed among cloud
Traffic	Raw data uploading	Extracted features
Analysis	Need further processing	Already processed
Privacy	Sensitive data	Irreversible data
Controllability	Controlled by provider	Controlled by user
Adaptability	Fully applicable	Available in 5G network

B. Opportunities in Edge Computing

In the era of big data and mobile computing, an extensive number of applications have emerged and become important solutions in a lot of areas, such as taxi sharing, mobile payment, social networks, etc. More and more services become computation-intensive and cloud-based, which, however, gives a heavy workload to both the network infrastructure and the cloud servers. As a good solution, edge computing becomes popular due to its ability to offload the computational load from central servers to the devices near the user end [8]–[15]. One of the most obvious advantages of edge computing is that it can empower the edge nodes with some essential computing abilities, which can provide lower response latency and better resource utilization rate. Another advantage is that it can balance the workload of the cloud servers [16], decreasing the possibility that they are overloaded. These features of edge computing structure can largely alleviate the aforementioned problem. Several instructive examples have been presented [17]–[25]. Bilal and Erbad [26] present an approach to improve users' experience of video generation and streaming with mobile edge computing. The authors work on the "edge based video generation" concept, and desire to implement a robust and efficient video generation and streaming approach for gaming interactive videos, multi-view videos, 360-degree videos, etc. The proposed method makes sure that the system can conduct the computation with the edge computing framework. Therefore, it can significantly improve the QoS of video service and give the users better experiences when using the edge computing based applications and services. Tran et al. [27] give another example using the edge computing approach for the video streaming. They use the edge servers to transcode the desired videos into several different versions, which have different resolutions or bit rates, and, therefore, differ in data size. This design can adapt the raw videos to various devices with different internet connection speed and video playing abilities.

C. Deep Learning for Crowdsensing

Although edge computing can partly solve the aforementioned problems in crowdsensing, another approach is needed to process the captured data, including the compression, encryption, and analysis. The compression is for the further decrease of communication cost, the encryption is used for protecting the users' privacy, and the analysis is essential to make the captured data into full play and utilization.

The compressed sensing [28] is a representative case for data compression. Based on a deep learning method, a binary autoencoders scheme is designed for compressed sensing, in which a binary sensing matrix and a recover solver are jointly optimized in the network training. Results show that the compressed sensing performs well in efficiency, and is preferable for real-time wireless applications.

Will deep learning revolutionize crowdsensing? [29] gives preliminary answers by implementing a prototyping deep learning engine on a mobile device SoC. Compared with other approaches in several typical crowdsensing tasks, the deep learning based methods show significant advantages on inference accuracy, without overburdening the mobile hardware.

Valerio et al. combine the deep learning with edge computing and crowdsensing-like tasks [30]. They consider three scenarios, i.e., calculating all the tasks on the local devices, calculating all the tasks on the remote cloud, or calculating the tasks on the devices and cloud at the same time. They model a network cost function to measure the approximate total cost for the deep learning computing. Given a desired accuracy for a specific task, their model can be used to compute the trade-off between performance and network cost and to find the minimum network cost.

Compared to the aforementioned deep learning based approaches, our method adopts a specially-designed hierarchical deep model, which can be well adapted to the edge computing framework. The difference between the proposed method with other deep learning based approach is shown in Table. I. Our method can solve many problems existing in the traditional approaches, such as the computing distribution, the communication cost, and the user controllability. The main differences between our approach and the existing ones include the following aspects. First, in the traditional solutions, the centralized servers take all the burden of computation task, while in the proposed approach, most of the computing resource in the cloud can be utilized, leading to a better balanced cloud environment. Second, in cloud-based approaches, each device sends the captured data to the centralized servers in the cloud, resulting in significant traffic to the network infrastructure, while in our method, only the extracted features are uploaded to the upper layer, instead of the raw data, which contains lots of redundant data. Third, as one major drawback of the existing approaches, the user devices have to upload the sensitive data to the centralized servers for further process, which may cause privacy leakage, while in this method, users can decide which level of abstraction can be uploaded. The higher level the abstractions are, the more security the user data has. However, there also have some disadvantages, for example , the network adaptability. Unlike the traditional methods which have few requirements on the network infrastructure and, therefore, are fully applicable on existing devices, the proposed method need the basic edge computing structure which needs upgrading on the cloud infrastructure, especially on the base stations. We will detail its methodology and implementation in the following sections.

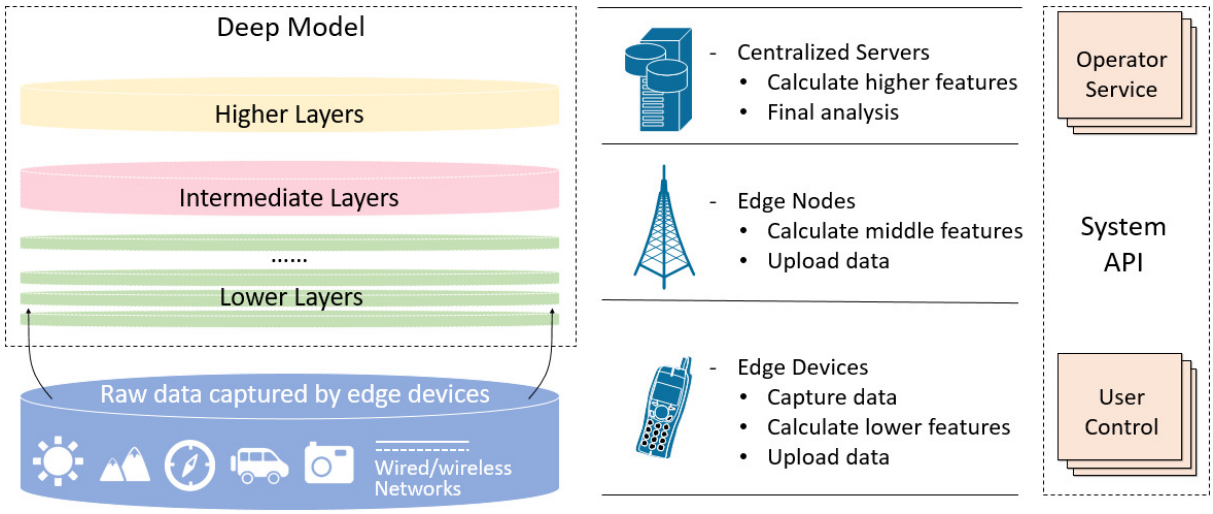


Fig. 2. The framework of the edge computing based deep learning system. The distributed deep model is allocated to all of the computing resource in the cloud, including centralized servers, edge nodes, and devices, for better load balance. Edge devices are responsible for the data sensing and lower layers calculation; the edge nodes calculate for the intermediate layers; the servers will perform the following higher layer calculation and final analysis. The crowdsensing process is fully controllable through some external APIs. Users can use the control panel to decide the balance between energy-saving and privacy protection.

III. Crowdsensing System: Concept and Design

In this work, we attempt to adapt the deep learning methods into the edge computing environment, with the following reasons. As mentioned above, edge computing is for the load balance of the cloud network, and there are two reasons we choose deep learning in the system. First, the crowdsensed data should ultimately be processed using some analysis methods, and deep learning is one of the most successful approaches to perform this task. Second, due to the hierarchical structure of deep learning models, it can well meet the requirement of edge computing. Deep learning is able to give both cost-efficiency and privacy protection to the crowdsensing system.

The main principle is to utilize the hierarchical deep model, and allocate its computation tasks to available resources in the cloud, including the centralized servers, the edge nodes, and devices. As shown in Fig. 2, an edge computing based deep (ECD) model is proposed for the crowdsensing task. ECD model is mainly characterized in its distributed nature and dynamic framework, which are both essential for the contemporary edge environment.

Fisrt, to adopt the deep learning in the edge computing area, the deep model must have the potentials for distributed operation. Same with all other deep models, the ECD model in the paper also has a large number of layers, as shown in Fig. 2. One obvious solution is to allocate the layers to all possilbe computing nodes. More precisely, the lower layers can be assigned to the edge of the cloud, and the higher layers can be assigned to the centralized servers. Because the activations, which are the output of each layer, require unidirectional transmission, each computing node must send the calculation results to the next logically adjacent node. The edge devices, including the mobile phones, deployed sensors, vehicles, and other devices which capture the raw data, are responsible for the first part of the crowdsensing task. As the nearest devices to the users, they should be exclusively under the control of their holders. According to the decision of the users, the edge devices can either calculate for a few lowest layers by themselves and send out the extracted features, or totally leave the calculation away and directly push the raw data to the network. It is a key feature of our human-driven design to empower the users with the power to make decisions, which will be detailed in the next section. The edge nodes, including cellular base stations, gateways, and routers, will handle the calculating of several intermediate layers, which is also a big difference with the existing deep learning based crowdsensing approaches. In the next generation of the network infrastructure, the edge nodes will be greatly enhanced in their computing ability, therefore, they will become an extremely important resource for cost-efficient services. The output of the edge nodes is uploaded to the central part of the cloud, i.e., the centralized server, for the calculation of higher layers and final analysis. Although the servers have powerful hardware to perform parallel computing, they are likely to be overloaded due to the extremely large sensing data captured by numerous device. However, with the feature extraction in the lower layers, the burden of centralized servers can significantly decrease.

Through the aforementioned one-way communication scheme, the ECD model obtains the basic distributing ability, however, it needs more to be fully adaptable to the edge computing architecture. The dynamic structure is another important piece of puzzle for this. Because one same deep model cannot handle all crowdsensing tasks, the ECD model in the proposed system should have the ability to be fitted into different objectives. On the other hand, the upgrade frequency of the edge hardware is usually very low, so it is very difficult to frequently change the lower-layer structures. Therefore, the proposed system is implemented with a flexible framework, which is adaptable for various crowdsensing tasks, while keeping the lower layers relatively stable. The lower layers are integrated into the firmware and only updated with a low frequency, and the higher layers are implemented as the

software installed in the centralized servers and keep adaptable for different tasks.

The service provider can deploy several sub-models with different lower layers for some typical kinds of crowdsensing tasks, such as the image classification, 3D scene understanding, road condition monitoring, audio recognition, network control [22], etc. A well-trained sub-model can be used for most of the subtasks in the same category, as their low-level features are very similar.

IV. Human-driven Crowdsensing

Following the trend of human-driven design, we attempt to empower the users with the right to decide the balance between energy consumption and privacy protection. On the one hand, users may prefer their devices can output more abstracted features, which is more secure but costs more energy; on the other hand, users may prefer to save the energy of their devices and leave the computing task to the outside devices, which can lead to some privacy concerns. The major reason is that the reconstruction from the output features to the original input deteriorates with the increase of the forward propagation progress. Therefore, there is a trade-off between efficiency and security. We believe, this dilemma should be decided, or at least partly directed, by the users. In other words, it must be guaranteed that the users can decide that to what level the deep model should be calculated on their own devices, which common users may not be very experienced in but have enough motivation to have the choice. With the human-driven design, service providers can give some recommendations or a changeable security level for the users, just like all the similar solutions integrated in mainstream operating systems or mobile applications for other security problems. As mentioned above, it must be guaranteed that the users can decide that to what level the deep model should be calculated on their own devices. Due to the characteristic of the deep learning model, the smallest calculation unit, which can be assigned to different computing nodes, is the layer. Therefore, the best way to control the calculation level is to select the desired layer numbers for calculation.

We model the aforementioned dilemma as an optimization problem, and define the solution set as $\mathcal{B} = \{b_1, b_2, \cdots, b_n\}$, where n represents the maximum number of the layers for calculation. As there are two main factors regarding the energy consumption of edge devices, i.e., the network communication $S_{b_i}^{comm}$ and deep model computation $S_{b_i}^{comp}$, we also define their energy consumption values as $P(S_{b_i}^{comm})$ and $P(S_{b_i}^{comp})$ respectively. In fact, given a specific model, both $P(S_{b_i}^{comm})$ and $P(S_{b_i}^{comp})$ have fixed values. A common deep model, shown in Fig. 3, is used as an example. The first few layers in this model are shown in the figure. The input data is image files with 1000×1000 pixels in three channels. The data column in the left part represents the output size of each layer, which is also the packet size for network communication. The ops column in the right part is for the unit operation number, which is a rough approximation of the calculation cost. To quantify the $P()$ function, we conduct several tests on various devices, and find the relationship between $P(S_{b_i}^{comm})$, $P(S_{b_i}^{comp})$ and

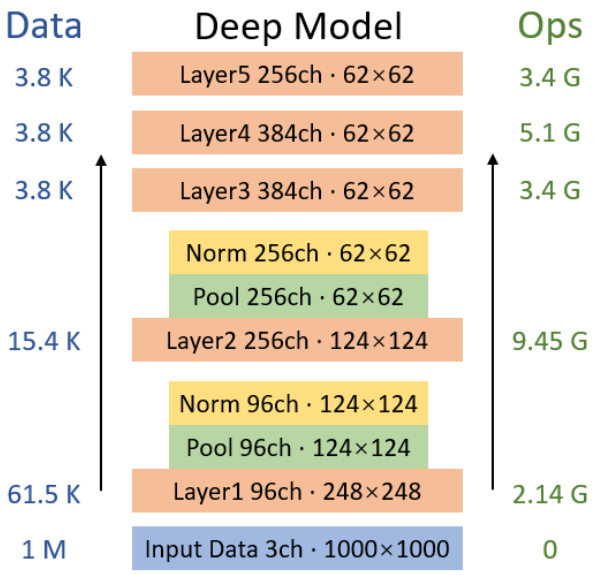
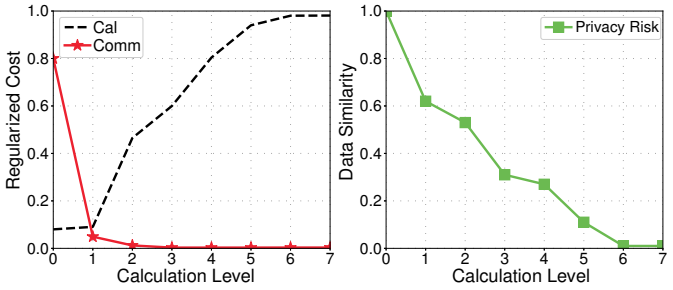


Fig. 3. The communication data size and the numbers of unit operations in a deep model.

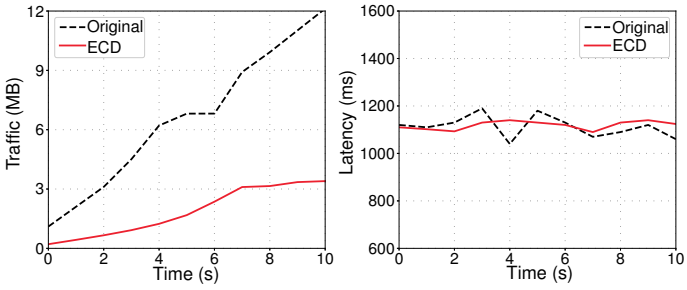
exact power consumption. As shown in Fig. 4(a), we define a regularized energy cost, in terms of Joule. The dotted black line represents the energy consumption resulted from the calculation, and the red line represents the communication cost. It can be seen the computing cost gradually increase with the calculation level, while the communication cost significantly drops down due to the decrease of output data size.

In addition to the energy consumption, the privacy protection $S_{b_i}^{priv}$ is another important factor to consider in our human-driven crowdsensing system. It is very difficult to accurately measure the risks of the possible privacy leak. A more obvious way is to calculate the consequences if the uploading data are captured by unauthorized devices. There is some approaches to reconstruct the input data with the output results, such as the intermediate features or even the final results. These approaches can start with a random generated noise data, calculate its output, and compare it with the objective feature, then these methods can optimize the generated input gradually, and, ultimately, get the data which maybe similar to the original input. Due to the characteristic of the deep learning models, it is not possible find the exact initial input from the output, and the more abstract the features are, the less similar the inferred input is. Therefore, we define a data similarity index $P(S_{b_i}^{priv})$ to measure the sensitivity of the output data. This index measures the degree of similarity between the reconstruction of the output feature and the raw captured data. Because the extracted features are not exactly reversible to the specific input, therefore, the output data with lower reconstruction similarity has a better security, and it can keep the users' privacy even if it is leaked to someone else. This point is especially notable in image processing tasks, where the activations of the higher layers are much more abstracted and illegible than the ones of the lower layers. In addition, there are lots of ways to measure the similarity, for example, the Euclidean distance, etc. Fig. 4(b) gives an



(a) (b)

Fig. 4. The energy cost and privacy risk of different calculation levels. (a)The calculation and communication cost of each calculation level. (b)The privacy risk of each calculation level. The risk is assessed by calculation the data similarity.



(a) (b)

Fig. 5. Network traffic and latency of the demonstration application.

example regarding the relationship between data similarity and calculation level. In conformance with our estimates, the similarity decreases with increasing calculation level.

Given $P(S_{b_i}^{comm})$, $P(S_{b_i}^{comp})$ and $P(S_{b_i}^{priv})$, an utility function is defined as

$$U = \lambda_1(P(S_{b_i}^{comm}) + P(S_{b_i}^{comp})) + \lambda_2 P(S_{b_i}^{priv}), \quad (1)$$

in order to make a comprehensive evaluation on different solutions. The weights λ_1 and λ_2 are set by the users to reflect the individual orientation. The solution with smallest U value is presented to the users as an instructive recommendation.

V. Performance Evaluation

A. Demonstration for System Validity

We first use a small testbed to demonstrate the validity of the proposed system, i.e., it is a feasible solution to divide the deep model into several parts and allocate them to different computational resources, and it works well with actual hardware. The testbed consists a central server, an edge server, and a mobile device. The edge server has a wireless network interface card for wireless access and an Ethernet network interface connected to the central server. We use the Raspberry Pi as the edge server, which is fully capable of forward propagation.

We implement one image classification application. The user captures an image, and attempts to know the information of the image. In our testbed, we install the classification model in the mobile phone, edge server, and the central server. The mobile device consistently captures images and calculates for the results using the available resources. We compare the performance between the common solution, i.e., directly uploading the raw data, and our method. The results are shown in Fig. 5. Fig. 5(a) gives the traffic change during actual deep learning based tasks. Compared with the original method, the proposed ECD approach consumes less network traffic. After 10 seconds, the traffic size of the original method have achieved three times larger than ours. In the meantime, according to Fig. 5 (b), the ECD can keep a similar performance with the original method in the response latency. It can be seen that the proposed method outperforms the original solution in the network traffic while keeping a similar latency.

B. Numerical Simulation

In this simulation, two servers, ten edge nodes and 100 mobile devices are deployed to serve as the edge computing system. The mobile devices keep performing sensing tasks and upload the data to the cloud. The adopted deep model is used for image recognition, and all the collected data is in the form of image. The data size is measured in megabyte.

As shown in Fig. 6, we test the energy saving performance with several different strategies, and present their consumption changes with increasing captured data size. The two black curves are the simulation results of the proposed ECD method, with $\lambda_1 = 0.8, \lambda_2 = 0.2$ and $\lambda_1 = 0.2, \lambda_2 = 0.8$, respectively. The green line represents the average strategy, i.e., always calculate for the average layers in the edge devices. The blue line is the maximum strategy, in which the edge devices are responsible for all lower layers; while the orange line represents the strategy in which the edge devices directly upload the raw captured data. And the last one, the red line, is a random select strategy, i.e., the calculation level is randomly selected in each run. It can be seen that the proposed ECD model has a significant advantage in energy efficiency. When λ_2 is set with a large weight value, the ECD model shows an obvious orientation on the cost saving. As a result, it achieves better performance than the one with a larger λ_1. On the contrary, when λ_1 is set to 0.8, the ECD model cares more about the user privacy. Fig. 7 (a) gives the comparison results of data similarity. And the deep model with a larger λ_1 value, "ECD2" bar in the figure, prefers safety over the energy efficiency. Of course, the "max" bar shows a huge lead in this test, because it calculates for all the lower layers and is able to output the most abstract features. In addition, we also present the privacy-protection efficiency in Fig. 7 (b), in order to measure the "value" of unit energy consumption. It can be seen, our method has the highest efficiency in improving the effect of privacy protection with one unit of energy consumption.

Another simulation is conducted to demonstrate the performance of the ECD model in balancing the computing load, as shown in Fig. 8. The difference between left and right subfigures is that the edge devices and edge nodes simply send the raw data to the centralized servers for further computing in the left, while in the right one both edge devices and nodes play a part in the computing task. The z axis represents the

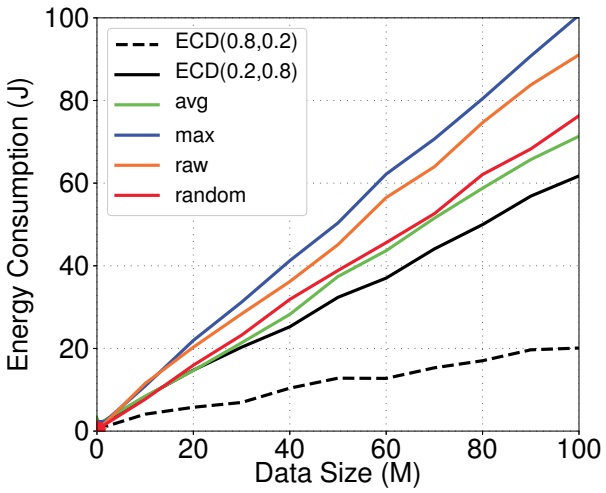


Fig. 6. The comparison results of energy consumption.

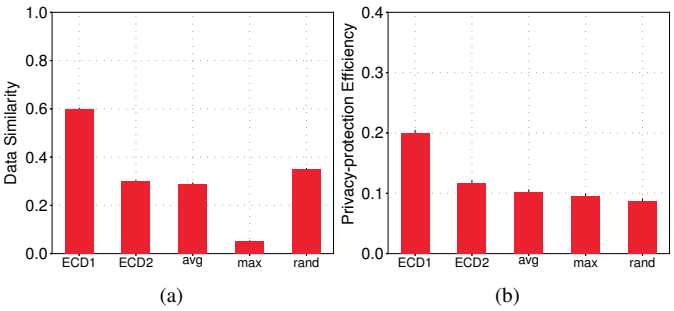


Fig. 7. The comparison experiments on privacy-preserving. (a)The results of output data similarity. (b)The results of privacy-protection efficiency.

hardware usage. We can see the servers in the left carry most of the calculation burden, which results in the steep surface in the left. On the contrary, with the ECD model, all the computing resource in the right can participate in the process of the captured data, and the load surface is flat and balanced.

These numerical results once again demonstrate the necessity and effectiveness of our hierarchical deep model. Compared to other approaches, the proposed ECD model performs much better in energy consumption, privacy security and load balancing.

VI. Conclusion

In this paper, we propose an edge computing based crowdsensing method, which can adopt the available computing resources in the whole network, both in the cloud and in the edge side, to ensure the load balance and reduce communication cost. We also adopt a specially-designed deep model to transform the crowdsensing problem into a hierarchical task, which is not only an effective data processing and analysis approach, but gives users the right to control the crowdsensing process. The experimental results well prove that it can provide better performance while considering and keeping users' privacy.

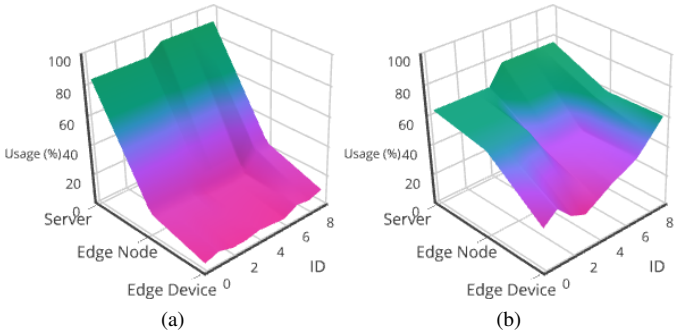


Fig. 8. Simulation results of load balance. (a)Edge devices directly upload the raw data for further process. (b) Edge devices and edge nodes participate in the pre-process of the captured data, using the proposed ECD model.

In the future, we need to test the compatibility of the proposed framework in different cloud environments, especially the calculation efficiency on various 5G base stations.

Acknowledgment

This work is partially supported by JSPS KAKENHI Grant Number JP16K00117, JP15K15976, and KDDI Foundation. Mianxiong Dong is the corresponding author.

References

[1] X. Zhang, Z. Yang, W. Sun, Y. Liu, S. Tang, K. Xing, and X. Mao, "Incentives for mobile crowd sensing: A survey," *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 54–67, Firstquarter 2016.

[2] P. P. Jayaraman, J. B. Gomes, H. L. Nguyen, Z. S. Abdallah, S. Krishnaswamy, and A. Zaslavsky, "Scalable energy-efficient distributed data analytics for crowdsensing applications in mobile environments," *IEEE Transactions on Computational Social Systems*, vol. 2, no. 3, pp. 109–123, Sept 2015.

[3] L. Pournajaf, D. A. Garcia-Ulloa, L. Xiong, and V. Sunderam, "Participant privacy in mobile crowd sensing task management: a survey of methods and challenges," *ACM SIGMOD Record*, vol. 44, no. 4, pp. 23–34, 2016.

[4] T. Luo, S. S. Kanhere, J. Huang, S. K. Das, and F. Wu, "Sustainable incentives for mobile crowdsensing: Auctions, lotteries, and trust and reputation systems," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 68–74, March 2017.

[5] H. Li, K. Ota, M. Dong, and M. Guo, "Mobile crowdsensing in software defined opportunistic networks," *IEEE Communications Magazine*, vol. 55, no. 6, pp. 140–145, 2017.

[6] L. Wang, D. Zhang, Y. Wang, C. Chen, X. Han, and A. M'hamed, "Sparse mobile crowdsensing: challenges and opportunities," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 161–167, July 2016.

[7] W. Sun and J. Liu, "Congestion-aware communication paradigm for sustainable dense mobile crowdsensing," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 62–67, March 2017.

[8] B. P. Rimal, D. P. Van, and M. Maier, "Mobile edge computing empowered fiber-wireless access networks in the 5g era," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 192–200, February 2017.

[9] X. Tao, K. Ota, M. Dong, H. Qi, and K. Li, "Performance guaranteed computation offloading for mobile-edge cloud computing," *IEEE Wireless Communications Letters*, vol. 6, no. 6, pp. 774–777, Dec 2017.

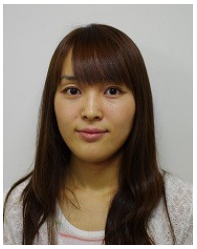
[10] J. Liu, J. Wan, B. Zeng, Q. Wang, H. Song, and M. Qiu, "A scalable and quick-response software defined vehicular network assisted by mobile edge computing," *IEEE Communications Magazine*, vol. 55, no. 7, pp. 94–100, 2017.

[11] C. Wang, F. R. Yu, C. Liang, Q. Chen, and L. Tang, "Joint computation offloading and interference management in wireless cellular networks with mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 8, pp. 7432–7445, Aug 2017.

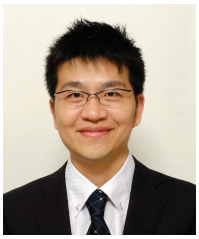
[12] S. Salsano, L. Chiaraviglio, N. Blefari-Melazzi, C. Parada, F. Fontes, R. Mekuria, and D. Griffioen, "Toward superfluid deployment of virtual functions: Exploiting mobile edge computing for video streaming," in *2017 29th International Teletraffic Congress (ITC 29)*, vol. 2, Sept 2017, pp. 48–53.

[13] S. Barbarossa, E. Ceci, M. Merluzzi, and E. Calvanese-Strinati, "Enabling effective mobile edge computing using millimeterwave links," in *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, May 2017, pp. 367–372.
[14] A. Kiani and N. Ansari, "Toward hierarchical mobile edge computing: An auction-based profit maximization approach," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 2082–2091, Dec 2017.
[15] K. Sato and T. Fujii, "Radio environment aware computation offloading with multiple mobile edge computing servers," in *2017 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, March 2017, pp. 1–5.
[16] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, "Collaborative mobile edge computing in 5g networks: New paradigms, scenarios, and challenges," *IEEE Communications Magazine*, vol. 55, no. 4, pp. 54–61, 2017.
[17] T. G. Rodrigues, K. Suto, H. Nishiyama, N. Kato, and K. Temma, "Cloudlets activation scheme for scalable mobile edge computing with transmission power control and virtual machine migration," *IEEE Transactions on Computers*, vol. 67, no. 9, pp. 1287–1300, Sept 2018.
[18] T. G. Rodrigues, K. Suto, H. Nishiyama, and N. Kato, "Hybrid method for minimizing service delay in edge cloud computing through vm migration and transmission power control," *IEEE Transactions on Computers*, vol. 66, no. 5, pp. 810–819, May 2017.
[19] F. Tang, Z. M. Fadlullah, B. Mao, and N. Kato, "An intelligent traffic load prediction based adaptive channel assignment algorithm in sdn-iot: A deep learning approach," *IEEE Internet of Things Journal*, pp. 1–1, 2018.
[20] B. Mao, Z. M. Fadlullah, F. Tang, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "Routing or computing? the paradigm shift towards intelligent computer network packet transmission based on deep learning," *IEEE Transactions on Computers*, vol. 66, no. 11, pp. 1946–1960, Nov 2017.
[21] Z. M. Fadlullah, F. Tang, B. Mao, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "State-of-the-art deep learning: Evolving machine intelligence toward tomorrow?s intelligent network traffic control systems," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2432–2455, Fourthquarter 2017.
[22] N. Kato, Z. M. Fadlullah, B. Mao, F. Tang, O. Akashi, T. Inoue, and K. Mizutani, "The deep learning vision for heterogeneous network traffic control: Proposal, challenges, and future perspective," *IEEE Wireless Communications*, vol. 24, no. 3, pp. 146–153, June 2017.
[23] M. Tao, K. Ota, and M. Dong, "Foud: Integrating fog and cloud for 5g-enabled v2g networks," *IEEE Network*, vol. 31, no. 2, pp. 8–13, March 2017.
[24] S. He, M. Dong, K. Ota, J. Wu, J. Li, and G. Li, "Software-defined efficient service reconstruction in fog using content awareness and weighted graph," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, Dec 2017, pp. 1–6.
[25] L. Li, K. Ota, and M. Dong, "Deep learning for smart industry: Efficient manufacture inspection system with fog computing," *IEEE Transactions on Industrial Informatics*, pp. 1–1, 2018.
[26] K. Bilal and A. Erbad, "Edge computing for interactive media and video streaming," in *2017 Second International Conference on Fog and Mobile Edge Computing (FMEC)*, May 2017, pp. 68–73.
[27] T. X. Tran, P. Pandey, A. Hajisami, and D. Pompili, "Collaborative multi-bitrate video caching and processing in mobile-edge computing networks," in *2017 13th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*, Feb 2017, pp. 165–172.
[28] B. Sun and H. Feng, "Efficient compressed sensing for wireless neural recording: A deep learning approach," *IEEE Signal Processing Letters*, vol. PP, no. 99, pp. 1–1, 2017.
[29] N. D. Lane and P. Georgiev, "Can deep learning revolutionize mobile sensing?" in *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*. ACM, 2015, pp. 117–122.
[30] L. Valerio, A. Passarella, and M. Conti, "Optimal trade-off between accuracy and network cost of distributed learning in mobile edge computing: An analytical approach," in *2017 IEEE 18th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, June 2017, pp. 1–9.

Liangzhi Li received the B.Sc and M.Eng degrees in Computer Science from South China University of Technology (SCUT), China, in 2012 and 2016, respectively. He is currently pursuing the Ph.D. degree in Electrical Engineering at Muroran Institute of Technology, Japan. His main fields of research interest include machine learning, big data, and robotics. He has received the best paper award from FCST 2017.



Kaoru Ota was born in Aizu-Wakamatsu, Japan. She received M.S. degree in Computer Science from Oklahoma State University, USA in 2008, B.S. and Ph.D. degrees in Computer Science and Engineering from The University of Aizu, Japan in 2006, 2012, respectively. She is currently an Assistant Professor with Department of Information and Electronic Engineering, Muroran Institute of Technology, Japan. From March 2010 to March 2011, she was a visiting scholar at University of Waterloo, Canada. Also she was a Japan Society of the Promotion of Science (JSPS) research fellow with Kato-Nishiyama Lab at Graduate School of Information Sciences at Tohoku University, Japan from April 2012 to April 2013. Her research interests include Wireless Networks, Cloud Computing, and Cyber-physical Systems. Dr. Ota has received best paper awards from ICA3PP 2014, GPC 2015, IEEE DASC 2015, and IEEE VTC 2016-Fall. She is an editor of IEEE Communications Letters, Peer-to-Peer Networking and Applications (Springer), Ad Hoc & Sensor Wireless Networks, International Journal of Embedded Systems (Inderscience) and Smart Technologies for Emergency Response & Disaster Management (IGI Global), as well as a guest editor of ACM Transactions on Multimedia Computing, Communications and Applications (leading), IEEE Communications Magazine, etc. Also she was a guest editor of IEEE Wireless Communications (2015), IEICE Transactions on Information and Systems (2014), and Ad Hoc & Sensor Wireless Networks (Old City Publishing) (2014). She was a research scientist with A3 Foresight Program (2011-2016) funded by Japan Society for the Promotion of Sciences (JSPS), NSFC of China, and NRF of Korea.



Mianxiong Dong received B.S., M.S. and Ph.D. in Computer Science and Engineering from The University of Aizu, Japan. He is currently an Associate Professor in the Department of Information and Electronic Engineering at the Muroran Institute of Technology, Japan. Prior to joining Muroran-IT, he was a Researcher at the National Institute of Information and Communications Technology (NICT), Japan. He was a JSPS Research Fellow with School of Computer Science and Engineering, The University of Aizu, Japan and was a visiting scholar with BBCR group at University of Waterloo, Canada supported by JSPS Excellent Young Researcher Overseas Visit Program from April 2010 to August 2011. Dr. Dong was selected as a Foreigner Research Fellow (a total of 3 recipients all over Japan) by NEC C&C Foundation in 2011. His research interests include Wireless Networks, Cloud Computing, and Cyber-physical Systems. He has received best paper awards from IEEE HPCC 2008, IEEE ICESS 2008, ICA3PP 2014, GPC 2015, IEEE DASC 2015, IEEE VTC 2016-Fall, FCST 2017 and 2017 IET Communications Premium Award. Dr. Dong serves as an Editor for IEEE Transactions on Green Communications and Networking (TGCN), IEEE Communications Surveys and Tutorials, IEEE Network, IEEE Wireless Communications Letters, IEEE Cloud Computing, IEEE Access, as well as a leading guest editor for ACM Transactions on Multimedia Computing, Communications and Applications (TOMM), IEEE Transactions on Emerging Topics in Computing (TETC), IEEE Transactions on Computational Social Systems (TCSS). He has been serving as the Vice Chair of IEEE Communications Society Asia/Pacific Region Meetings and Conference Committee, Leading Symposium Chair of IEEE ICC 2019, Student Travel Grants Chair of IEEE GLOBECOM 2019, and Symposium Chair of IEEE GLOBECOM 2016, 2017. Dr. Dong was a research scientist with A3 Foresight Program (2011-2016) funded by Japan Society for the Promotion of Sciences (JSPS), NSFC of China, and NRF of Korea. He is the recipient of IEEE TCSC Early Career Award 2016, The 12th IEEE ComSoc Asia-Pacific Young Researcher Award 2017.