2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovations

Real-Time Data Processing Architecture for Multi-Robots Based on Differential Federated Learning

Wei Zhou[1], Yiying Li[1], Shuhui Chen[1], Bo Ding[1]
College of Computer, National University of Defense Technology[1]
ChangSha, China

***Abstract*—The emergency of ubiquitous intelligence in various things has become the ultimate cornerstone in building a smart interconnection of the physical world and the human world, which also caters to the idea of Internet of Things (IoT). Nowadays, robots as a new type of ubiquitous IoT devices have gained much attention. With the increasing number of distributed multi-robots, such smart environment generates unprecedented amounts of data. Robotic applications are faced with challenges of such big data: the serious real-time assurance and data privacy. Therefore, in order to obtain the big data values via knowledge sharing under the premise of ensuring the real-time data processing and data privacy, we propose a real-time data processing architecture for multi-robots based on the differential federated learning, called RT-robots architecture. A global shared model with differential privacy protection is trained on the cloud iteratively and distributed to multiple edge robots in each round, and the robotic tasks are processed locally in real time. Our implementation and experiments demonstrate that our architecture can be applied on multiple robotic recognition tasks, balance the trade-off between the performance and privacy.**

Keywords—Real-time, ubiquitous intelligence, multi-robots, federated learning, differential privacy

I. INTRODUCTION

In the past decade, we have witnessed the immeasurable success of Internet of Things (IoT) [1] and it has enabled the realization of smarter planet, physical infrastructure and software services that enhance the quality of people's lives. Here, "Things" is an active and intelligent participant in the physical world that fully communicates and interacts with the real physical world through the exchange of data with the environment [2]. At present, a series of novelty ubiquitous devices (i.e., robots), mushroom in many areas of the real world, such as health, household, industry, agriculture, military [34]. With the increasing number of distributed multi-robots and complexity of robotic tasks, such environment could generate unprecedented amounts of data which may overwhelm today's analytics applications and storage systems.

In the robotic environment, robots need to address the challenges of big data. Many types of robotic tasks are data-intensive, such as SLAM (Simultaneous Localization and Mapping) [11], and so forth. Such challenges can be mainly divided into two aspects: the challenge of real-time data processing due to robots being closely related to the physical world; the challenge of privacy protection due to robots being closely associated with the human world.

Given the challenges of real-time requirement and privacy protection, a simple approach is that each robot handles its own data locally. However, it will definitely increase the local burden of robots on their limited computing resources. More importantly, such solution could not achieve the knowledge sharing effectively and result in an isolated information island. Knowledge sharing [12, 13] is of vital importance to produce the business values which will boost robots' autonomy. One of the essential attractions is to aggregate the big data and infer the business values by performing data analysis instead of data isolation.

Nowadays, with the rapid development of cloud computing, some researchers have proposed "cloud robotics" [3], aiming to provide scalable cloud storage and superior services on demand so that robots can break their own resource constraints, as shown in Figure 1. In addition, the knowledge sharing can be taken based on the cloud global platform. However, the network performance has become a major bottleneck in cloud-based architectures. We look forward to the future 5G technology can bring higher quality network services to mobile devices such as robots. Nevertheless, today's most ISPs only tout download speeds, while in the cloud-based age, the overshadowed upload speed has become critical to business operations. As the network test in [5] on 25 Nov 2017, the download speed is 20Mbps and the upload speed is only 5Mbps over a Wi-Fi connection. For robotic emergency response, auto-driving, and other latency sensitive robotic applications, such delays communicating with cloud is intolerable, for robots directly act upon the physical world and their applications give much attention to the quality of service (QoS), such as real-time assurance [6]. For example, self-driving cars generate 1 Gigabyte of comprehensive data per second, so the data processing needs a strong real-time assurance [7], or a disastrous traffic accident could be caused. (Different from some other ubiquitous devices like mobile phones, where a long response time may only cause a bad user experience.) In addition, since robotic raw data is exposed in the network, the robotic data privacy security is difficult to be guaranteed.

Therefore, in the robotic environment, how to obtain the shared knowledge business values under the premise of ensuring real-time data processing and data privacy is worth serious thinking. In this paper, we propose a real-time data processing architecture for multi-robots based on differential federated learning, called RT-robots architecture. Here, the federated learning [8] architecture includes a shared global model on the cloud trained under the coordination of a central server, from a federation of participating edge devices (i.e., multi-robots). Then the trained shared model is sent back to

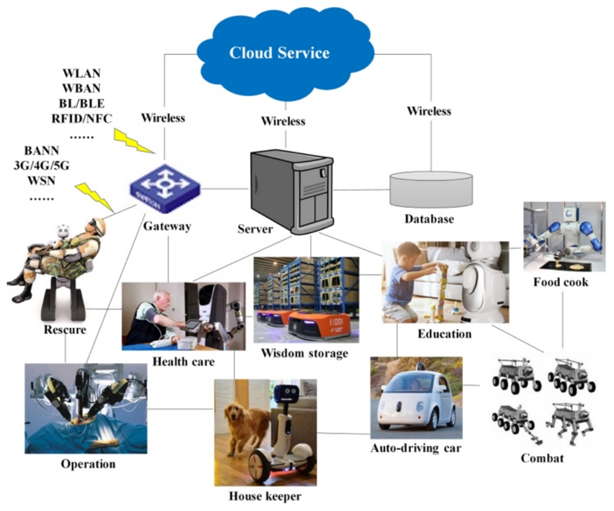

Fig. 1. A paradigm of multi-robots environment.

each robot in every round. In the application stage, robots' sensed data can be analyzed locally to satisfy the real-time requirement without the unpredictable network transmission overhead and robots can benefit from the shared knowledge values. Moreover, instead of uploading the raw robots' data to cloud, each robot only sends the gradient parameters updates to fine-tune the shared model based on data differential privacy [9] processing. In the experiment, we apply our architecture to a study case, robotic recognition, to illustrate the advantages.

The remainder of this paper is organized as follows. The background and related work are discussed in Section II. Section III presents the overview of our real-time data processing architecture for multi-robots and focuses on a study case. Section IV looks into the key technologies and analysis of the real-time differential federated learning architecture in depth. Section V describes the experiments to evaluate our work. Finally, we present our conclusion in Section VI.

II. Background and Related Work

This section first introduces the characteristics of big data business values in robotics, taking the recognition task in robotics as an example. Then, we discuss the necessity and the existing solutions to achieve real-time assurance in IoT environment. Finally, we introduce the vital technologies related to our work about the federated learning and differential privacy.

A. Big Data Business Values in Robotics

In the robotic environment, robots interact directly with the physical world, so being able to recognize their surroundings accurately is a fundamental capability to accomplish complex tasks. We take the robotic recognition [10] as a study case to talk about the business values on such big data. There have been various object recognition services on the Internet that can be used in robotics based on the collected large amounts of data, such as CloudSight [14], Google Vision API [15], and Image++ [16]. Moreover, in the way of knowledge sharing, a kind of business value brought by the "robots learn by each other" has gained much attention. A typical example is the "Million Object Challenge" Project [17] in Brown University. It aims to make robots around the world learn how to find and handle simple items based on their shared knowledge, for example, recognizing and picking up a certain object, while robots upload data to the cloud for training and allow other robots to analyze and use the data and information. Another example is robots could learn to recognize the food and make breakfast by gathering information on multi-robots to an online database called "Open-Ease" [18]. All in all, the performance of multi-robots could be boosted in the knowledge sharing mode based on a centralized cloud.

B. Solutions to Real-time Data Processing

Many robotics tasks ask for the real-time assurance, and a result has to be returned to the robot in time. As mentioned before, the delays caused by data processing mechanisms based on cloud requests service are intolerable due the poor network connections such as network congestion or even packet loss. To deal with this problem, edge computing [7] is proposed to make full use of the nearest computing and storage resources of ubiquitous devices for storage and simple data processing. This would release network pressure and accelerate decision making. However, as more and more IoT applications emerge, the limited resources of the edge near devices are competed, which results in resource preemption and increasing processing latency. Therefore, fog computing [4] is then introduced. It seamlessly integrates edge devices and cloud resources by cooperating the use of distributed edge resources and cloud. IoT applications such as robotic tasks would be assigned or allocated dynamically to the edge or cloud based on the requirements like the latency-sensitivity level of tasks or in time sequence. Different mechanisms can be designed in fog computing paradigm based on different tasks. Inspired by the coordination idea of fog computing, we propose a novel architecture for multi-robots to meet the requirement of obtaining real-time assurance while utilizing the knowledge sharing values. Different from the work of fog computing [4, 19], our work makes full use of the data analysis abilities of edge robots themselves and the cloud resources. In addition, we put forward a federated learning based real-time data processing architecture clearly and exactly to achieve the synergy between edge devices and cloud, and apply this architecture into a robotic recognition application.

C. Federated Learning and Differential Privacy

Federated learning [8] is originally a machine learning setting proposed by Google in 2016, where the goal is to train a high-quality centralized model while training data remains distributed over a large number of clients each with unreliable and relatively slow network connections. We consider this computing paradigm of federated learning is suitable for data analysis in the environment we are considering. In each round, each device client independently computes an update to the current model based on its local data, and transmits this update to a central cloud server, where the device client-side updates are aggregated to compute a new global model. As a result, the big data generated by distributed devices can achieve knowledge integration rapidly and the shared model is then distributed to each device for application in real time. However, though the local private data no longer needs to be exposed in the link and cloud, the relative updated parameters and shared model in the latest round may be tapped. As the analysis and argument in [20], a successful deep network can even rigorously recover or fit the original train data. We do not want the edge devices like robots in people's daily life to leak users' privacy in the knowledge sharing process. Therefore, we also introduce the differential privacy technology to the federated learning based architecture in the parameters updates phase to provide more protection without sacrificing the knowledge values. Differentiated privacy [9,21] solves the paradox of hiding local private samples while learning useful information about global samples. It is a privacy definition tailored to privacy-protected data analytics issues by mixing noise with data or in model processing phase. However, A trusted curator aggregates parameters optimized in decentralized fashion by multiple clients [36]. Unlike this assumption of trust, our central cloud is only responsible for the aggregation of the global model, and the privacy protection process is done on the client side, this is more in line with decentralized privacy protection.

III. RT-ROBOTS ARCHITECTURE OVERVIEW

The RT-robots data processing architecture is designed to fulfill three goals: (1) Utilizing the cloud for building the shared model incrementally based on data generated on multi-robots, and (2) Protecting the privacy and safety of each robot while realizing the knowledge sharing, and (3) Making full use of each robot's ability for real-time recognition in the application stage. The first one is realized by taking advantage of the powerful computation capability and the centralized aggregation platform of the cloud in the continuous training stage and introducing a state-of-the-art PCA [22] or CNN-based classification model. As to the second one, we adopt the technology of differential privacy on each robot in the training stage to ensure the robot's local data privacy. As to the third one, we distribute the shared model in each round to edge robots for the model synchronization, and then robots could recognize objects locally in real time for better environment exploration. In this paper, we consider the work of knowledge sharing incrementally on cloud and shared model application locally as the iterative federated learning process.

Figure 2 gives an overview of the main components of RT-robots data processing architecture. We build the architecture based on the network framework of 5G and LoRaWAN [35]. 5G is mainly used for multimedia data transmission, and LoRaWAN is responsible for controlling the channel. There are two roles in this system:

Edge Robots. Robots are autonomous devices that move, sense and take actions in various environment. Robots in different places may encounter different objects instances continuously. In the training stage, robots may get new objects samples with labels given by their hosts incrementally. For example, the owner would take the initiative to tell the robot "it is a new table though its shape is different from the old one", or the robot can figure out the objects labels and attributes by reading the objects brands. Therefore, in this case, we think about the issue that "how to improve the environment recognition abilities of the entire robot community based on

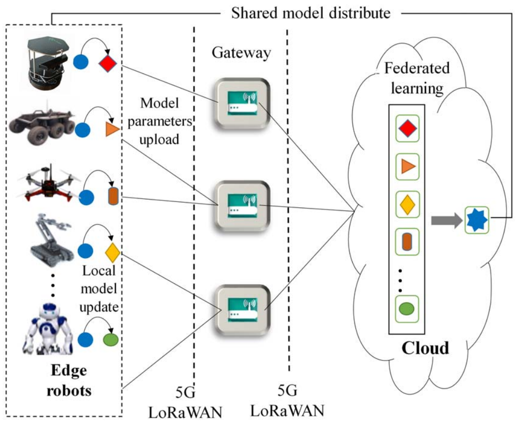

Fig. 2. The RT-robots architecture overview.

their sensed data?" Then in the application stage, when another robot encounters that new table, it can recognize it correctly based on the shared knowledge instead of asking its owner. In addition, we hope to learn the common and general objects knowledge without leaking the privacy of each robot data. Therefore, each robot has these three modules: (1) Local model update with differential privacy module; (2) Model upload and reception module; (3) Real-time recognition module.

Cloud. Cloud is a server deployed on a data center with powerful computing and storage capabilities. It has a global shared model and the model is updated in every round. The cloud is composed of two modules: (1) Models collection and distribution module; (2) Federated learning module. The cloud adopts a PCA or CNN-based approach to achieve the feature extraction and the softmax classifier outputs the object category labels as the classification results.

At runtime, the robots' performance is boosted continuously. At the beginning, the cloud global model is initially trained by the large amount of public data (the confessed data that can be touched by anyone) and distributed to each robot in the environment. Each robot interacts with its surroundings and collects the private raw data (data do not want to be obtained or recovered by other ones) continuously and then computes an update to the local current model with its idle computing power. In each round period, each robot regularly updates its model parameters to the cloud server after the differential privacy processing. Then the cloud collects the multiple current model updates from multi-robots and then the federated learning module learns an updated centralized model federally based on the "model averaging" approach [23]. After the global model is updated, it is shared and distributed to each robot for the following real-time recognition application locally. Such round is taken iteratively and continuously.

As to the real-time requirement of our RT-robots data processing architecture, the main advantage is that we move the task down to the robots themselves for object recognition based on the distributed shared model. Different from the traditional cloud-based architecture that robots need to transmit the collected data to the cloud and wait for the result response, our approach gets rid of the dependence of the cloud and unpredictable network (A big latency can be caused by the discarded services, the network fluctuations, the cloud queuing and so on.) in the application stage. Moreover, as to a recognition model, it is the training stage that takes up more computing power instead of the application stage. The cloud is responsible for knowledge integration by model averaging and the robots only train the current model when the computing equipment is idle. As long as an object image needs to be recognized in a robot, the recognition task has the top priority to be dealt with based on the robot's current local model. A real-time theoretical analysis and experiments are carried out in Section IV and V. In this paper, we do not consider the negative impacts on real-time assurance brought by the storage in robots, and the local private data can be dumped to other trustworthy storage devices or discarded depended on the users after the local current model updates.

IV. Key Technologies of Differential Federated Learning with Multi-Robots

In this section, we introduce the details of the differential federated learning algorithm with multi-robots in depth, which is illustrated in Figure 3, and gives the theoretical analysis of the architecture performance benefits. The technologies in our differential federated learning algorithm include: how to use the public data and robot's private data for knowledge integration, how to use the noise aggregation to achieve privacy protection, and how to fine-tune the hyper-parameters of the model.

A. *Differential Federated Learning Algorithm*

Each robot's execution environment has some certain regional characteristics, and according to the differences of the robots' tasks, robots will produce a large number of private local data. In our work, we want to take advantage of the value of robotic private data to enhance our overall knowledge. In this way, multiple robots can overcome the limitations of local knowledge and address the challenges of an open environment. At the same time, as embedded device performance evolves, we want to take full advantage of the computational resources of a single robot for the edge analysis. Therefore, we take a federated learning approach on the cloud to achieve the expansion of knowledge and efficient task collaboration, so as to improve the abilities to deal with a wide range of specific tasks under the environment. At the same time, pure multi-robots cooperative training to a certain extent will cause the risk of disclosure of private data. Some existing distributed federated learning approaches ignore privacy restrictions. Here, we introduce the differential privacy processing on each robot. Differential privacy can ensure that a rival cannot observe the output information to learn the individual information in the knowledge model. In this way, it is possible to enhance the robots' data processing performance, while also protecting the data privacy of each robot.

Usage of private and public data. As mentioned before, the global model is initialized on the cloud based on the public data. Robots have their own private local data and can have the access to all the public data. Each robot's local private data is independent with each other, and we want to make full use of each robot's private data to improve the overall performance.

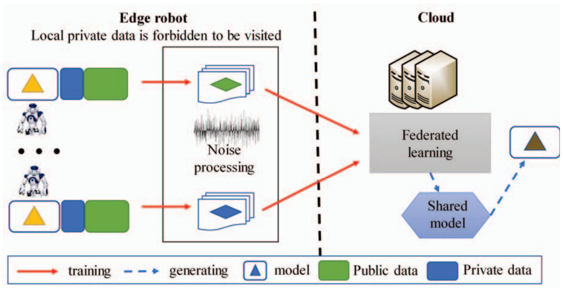

Fig. 3. The RT-robots architecture overview.

To protect the privacy in the meantime, we take noise processing to the private data. In addition, in order to avoid the possible negative impact on the accuracy brought by the noise, we construct the training data that contains not only robot's private data but also the public data. Assume the whole recognition dataset in the robots community is (X,Y), where X represents data and Y represents the corresponding labels. n represents the number of robots. $Robot\_private_i$ refers to the private dataset on robot i. Therefore, we give the representation of multi-robots datasets as the following.

$$Robot\_private_i = (x_{pri\_i}, y_{pri\_i}) \tag{1}$$

$$\begin{aligned}(X,Y) &= (X_{pri}, Y_{pri}) + (X_{pub}, Y_{pub}) \\ &= \sum_{i=1}^{n} Robot\_private_i + public\_data\end{aligned} \tag{2}$$

Parameters processing with differential privacy. In order to make use of the robots' computing resources efficiently, the model parameter updates are figured out and transferred to the cloud instead of the whole models. For privacy on each robot, we iteratively calculate the gradient of randomly sampled data (as a mini-batch) and add noise to the calculated gradient in each round instead of the robot's final model of the round, because adding noise to the robot's model would destroy the utility of the learned model. Here, a "round" refers to the procedure that each robot calculates its model gradient based on one mini-batch with noise and uploads to the cloud, followed by the cloud updates its global model and distributes to each robot. In addition, the "parameter update" in this paper exactly refers to the gradient with noise. We will give the details of the parameters processing with differential privacy as the following.

Differential privacy [24, 25] can provide strong privacy guarantees for multi-data aggregation. It is defined in terms of the concept of adjacent dataset. "Adjacent" is defined as: there is only one sample difference between the two datasets.

Definition 1. A randomized mechanism $M: D \to R$ with domain D and range R satisfies (ε, δ) -differential privacy if for any two adjacent inputs d, d' and for any subset of outputs $S \subseteq R$, it holds that

$$\Pr[M(d) \in S] \le e^{\varepsilon} \Pr[M(d') \in S] + \delta \tag{3}$$

We learn from the work introduced by Dwork et al. [26], which allows for the possibility that plain differential privacy is broken with probability δ. For instance, we refer to the work of Martín et al. [29], and set the basic definition of the mapping function by

$$M(d) \triangleq f(d) + N(0, S_f^2 * \sigma^2) \tag{4}$$

The noise here is a normal distribution that meets the standard deviation $S_f * \sigma$. Also, there are some other noise processing technologies can be used, such as Papernot et al. [21] uses another type of noise $Lap(\frac{1}{\gamma})$, γ is a privacy parameter and $Lap(b)$ the Laplacian distribution with location *0* and scale b.

We consider the problem of machine learning optimization of multi-robots, and calculate the loss function of each sample in a single robot. Suppose there are n robots and the size of all dataset in the whole robot community is m. In practice, we perform the mini-batch stochastic gradient descent (SGD) algorithm on each robot. $Robot_i$'s mini-batch contains random samples from both private and public dataset $(Robot\_private_i + public\_data)$. In every round, the sample dataset in $Robot_i$'s mini-batch is defined as $batch_i$, the ratio of private data in the sample data is α, and the number of its samples is $m = |batch_i|$. So, the number of all samples on all robots in one round is $M = \sum_{i=1}^{n} m_i$. Therefore, our objective function is

$$f(\theta) = \sum_{i=1}^{k} \frac{m_i}{M} f_i(\theta) \tag{5}$$

$$f_i(\theta) = \frac{1}{m_i} \sum_{j \in batch_i} f_{ij}(\theta) \tag{6}$$

Algorithm 1 outlines our basic method for differential federated learning algorithm. For each robot, we use the model parameters θ_t at round t to train the gradient $\nabla L(\theta_t, batch_i)$ of random sample data $batch_i$ and then add noise $N(0, \rho_i)$ to the gradient of private sample data to ensure privacy.

$$g_i(t) = \nabla L(\theta_t, batch_{private}) + N(0, \rho_i) + \nabla L(\theta_t, batch\_public_i) \tag{7}$$

The above process is executed on each robot, and then the cloud collects the multiple model parameter updates. What needs to be emphasized here is that the noise distribution of each robot is not the same. Finally, the cloud updates the model after calculating the average of all the parameter updates, and releases the new model to all the robots after a certain period.

Hyper-parameter fine-tuning. In the recognition neural network, PCA or CNN is used for the feature extraction of the samples, PCA has a good speedup function compared to CNN, and CNN has a remarkable effect in the feature extraction. In order to ensure the trade-off between accuracy and privacy, we will adjust hyper-parameter of the network. Through experimental observation, we found that the accuracy of the model will be affected by the batch size, noise distribution and the number of robots.

Algorithm 1: Differential Federated Learning

Input: dataset $D = (X, Y)$ noisy parameter ρ, the number of robots n, shared model parameters θ

Output: the shared model and evaluation accuracy

```
1  Initialize Split dataset $D$ according to the number of robots
   and public data, $D = \sum_{i=1}^{n} Robot\_private_i + public\_data$
2  for $t \in$[max_round] do
3      Each robot computes its gradient
         For each $Robot_i$, take a random sample data from
         $(Robot\_private_i + public\_data)$
         with sampling ratio $\alpha$, compute gradient
             $g_i(t) = \nabla L(\theta_t, batch_{private}) + N(0, \rho_i) + \nabla L(\theta_t, batch\_public_i)$
4      Cloud convergence gradient
             $g_t = \frac{1}{n}(\sum_{i=1}^{n} g_i(t))$
5      Descent
             $\theta_{t+1} \leftarrow \theta_t - \tau g_t$
6      Broadcast parameters
         For each $Robot_i$, receives model parameters $\theta_{t+1}$
7  end for
8  Output the results
       $\theta_{\max\_round}$ and calculate the evaluation accuracy
```

In the setting of hyper-parameter reference, we take experiments to try and learn from the expert experience. Our ultimate concern is to ensure the accuracy of the model under the premise of privacy.

B. Performance Analysis of RT-robots Architecture

The benefits of RT-robots data processing architecture can be analyzed theoretically from the following three aspects:

1) Real-time assurance. In the introduced RT-robots architecture, the cloud is responsible for the federated learning process, and the robots are responsible for edge tasks applications. Compared to the request & response-type service, our approach dramatically reduces network transmission latency and cloud server service queue latency.

The network transmission delay is related to the extent of path congestion. In order to simplify the communication model, we introduce the queuing theory to analyze the network communication process, to estimate the network end-to-end latency based on the M/D/1 queue model [27]. End-to-end latency includes network latency (d_{net}) and cloud server processing latency (d_p), d_p is calculated based on the specific tasks. What we are concerned about here is the network latency consisting of queuing latency (d_q) and propagation latency (d_{prop}), in which the queuing latency is determined according to M/D/1 theory and the propagation latency is set according to the experience value. The total delay is

$$d = d_p + d_q + d_{prop} \tag{8}$$

For M/D/1 theory, the queuing latency of a single robot is calculated as follows:

$$d_q = \frac{\lambda}{2\mu(\mu-\lambda)} + \frac{1}{\lambda} \quad (9)$$

λ is the average arrival rate of the robot to reach the cloud, and μ is the task queue capacity of the cloud server.

2) Accuracy improvement. In a traditional multi-robots environment, there is a certain amount of public dataset for a particular recognition task. Our architecture assumes that apart from exposing datasets, each robot owns private data that is invisible to each other based on privacy protection. Third-party servers cannot collect private data from all robots. This creates an obstacle to training a shared model.

According to a large number of previous research results, we can find out the performance grows logarithmically as pre-training data expands [28].

Based on the various tasks of machine learning, the model interpretation ability is directly proportional to the size of the tasks dataset. Therefore, we design RT-robots data processing architecture, in the case of privacy protection, to enhance the performance of the system to a certain extent. Of course, such performance includes the accuracy.

3) The moments accountant of privacy loss. Adding noise for privacy protection will affect performance to a certain degree. We hope to quantify the impact of differential privacy protection on the model through the statistics of cumulative privacy losses in the whole training process. The following variant of differential privacy introduced in [24].

Definition 2. A randomized mechanism $M : D \to R$ with domain D and range R satisfies (ε, δ) -differential privacy if for any two adjacent inputs $d, d^{'}$, aux is an auxiliary input. For an outcome $o \in R$, the privacy loss at o is calculated as:

$$c(o; M, aux, d, d^{'}) \triangleq \log \frac{\Pr[M(aux, d) = o]}{\Pr[M(aux, d^{'}) = o]} \quad (10)$$

The privacy loss random variable $C(M, aux, d, d^{'})$ is defined as $c(o; M, aux, d, d^{'})$. For the calculation of cumulative loss, we adopt the recent advances called moments accountant. The moments accountant is the calculation theory that designed by Abadi et al. [29] on the basis of [24].

Definition 3. A randomized mechanism $M : D \to R$ with domain D and range R satisfies (ε, δ) -differential privacy if for any two adjacent inputs $d, d^{'}$, aux is an auxiliary input. The moments accountant is calculated as:

$$\alpha_M(\lambda) \triangleq \max_{aux, d, d^{'}} \alpha_M(\lambda; aux, d, d^{'}) \quad (11)$$

$$\alpha_M(\lambda; aux, d, d^{'}) \triangleq \log E[\exp(\lambda C(M, aux, d, d^{'}))] \quad (12)$$

The other characteristics about the nature of the moments accountant can refer to [29].

V. Experiment and Evaluation

In the experiment settings, we distribute the cloud and simulation robots in different locations, the cloud is deployed on an IBM x3660 M4 server which is equipped with a NVIDIA GTX 1080 Ti, and the 100 simulation robots are composed of compute nodes in different locations. We take the experiments on two popular image datasets: MNIST [30] and CIFAR-10 [31] and implement our RT-robots data processing architecture on the latest TensorFlow [32]. The recognition model is realized by PCA(Principal Component Analysis) or CNN(Convolutional Neural Network) feature extraction approach. We focus on the real-time assurance of RT-robots data processing architecture on the edge. In addition, the trade-off performance of accuracy and privacy is greatly dependent on the injection of noise.

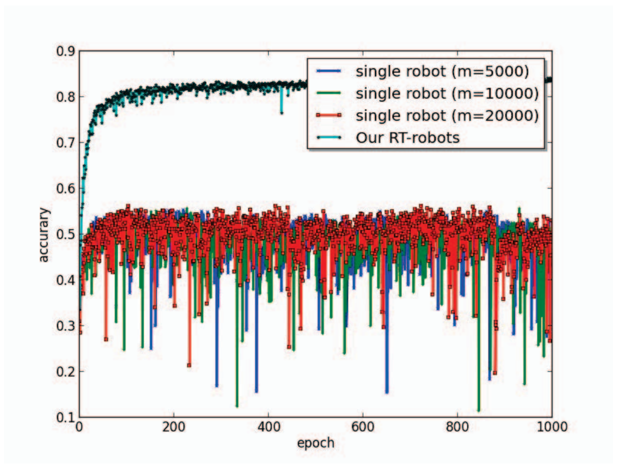

Fig. 4. Comparison of our RT-robots and single robot architecture.

In order to reflect the advantages of federated learning in big data, we compared the performance differences between robotic recognition tasks using a federated learning approach and not taking this approach. The existing robot recognition task process adopts the mode of offline stand-alone training, lacks the support of robot big data resources and full use of the multi-robot computing ability. The purpose of federated learning is to obtain greater training set support, assuming that the robot has private data. As shown in Figure 4, when building a multi-robot federated learning framework, we could get a significant improvement on the model accuracy and it maintains at a relatively high and stable level. Here, one epoch refers to 100x rounds to better show the results, m represents the size of a single machine private data set.

A. MNIST

We evaluate the experiments on the standard MNIST dataset for handwritten digit recognition consisting of 60,000 training examples and 10,000 testing examples. Each example consists of a 28 * 28 grey image and a label with the number marked out. Our MNIST model stacks two convolutional layers, one fully connected layer and one softmax layer of 10 classes with cross-entropy loss. As to the CNN-based approach, it could extract complex and high-dimension features, which may call for more computation and we pre-train the convolutional model for speeding up. PCA could extract dimension reduction features efficiently and directly, for the MNIST images are simple and its input vector is relatively sparse. In this part, the convolutional layer can also be replaced by a PCA structure for feature extraction.

We assume that the dataset contains private data and public data, and take into account the scarcity of private data for each robot. We set 10,000 samples for the public data, and 100 robots hold the remaining 50,000 samples as private data on average. Each robot uses its own privacy data and public data, enhancing the overall ability of the task under the federated learning. The global model trained in this way is distributed on each robot to meet the requirements of real-time, while making full use of the computing resources and privacy data of each robot.

1) Recognition latency optimization. In the experiments, a large number of robots and the cloud directly connect via Wi-Fi. We consider the latency comparison of these two cases: (1) Each robot is only responsible for collecting data, and then sending data to the cloud through the link to request for recognition result. The cloud figures out and returns the result data. (2) Our approach: the global model is on the cloud and distributed to the robots periodically, and each robot figures out the result on the edge. At the same time, the cloud uses multi-robots parameter updates to complete the federated learning using their privacy data.

As to the selective PCA or CNN feature extraction operations, Figure 5 shows the results of the recognition latency on a single robot. It shows the current results of the mnist_PCA latency and mnist_CNN latency as the number of rounds increasing. On average, the mnist_PCA latency is about 2.5ms with fluctuation. In contrast, the delay of convolutional operation is larger, about 7ms. Since the characteristics of the MNIST data sample are relatively simple, the PCA operation can effectively extract the main features of the sample, and it computes a mapping of 784 to 60 dimensions, which can efficiently reduce the sampled data dimensions. However, CNN has a large number of convolution kernel operations, and its computation is relatively large. Therefore, we can find a significant latency difference between the two operations.

In order to highlight the real-time advantages of the edge data process, we contrast it with the traditional way (request & response-type with the cloud) and the experiments are conducted to evaluate the effects of variables related to the edge processing mechanism. Figure 6 presents the end-to-end service latency with the variable of the number of concurrent robots. Meanwhile, we conducted a horizontal comparison with the single-robot case. Both single robot case latencies are below 10ms, the latency in the way of requesting service from the cloud is higher, about 30ms. Here, the variable is the number of concurrent robots, and it can be seen that as the number of concurrent requests increases, the service delay is also slowly increasing. The main reason is: according to M/D/1 theory, the server processing capacity is determined, the queuing latency will gradually increase with the number of concurrent visiting robots.

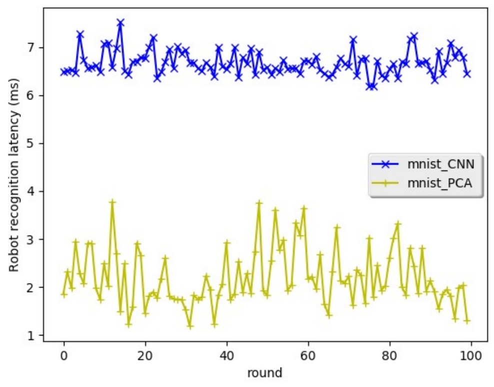

Fig. 5. The recognition latency on a single robot.

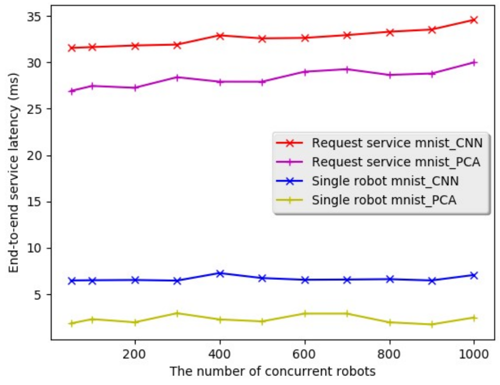

Fig. 6. End-to-end service latency.

2) Experiments of accuracy and differential privacy in federated learning model. First, we distribute the dataset to different robots in a certain proportion, and take the experiments without taking the differential privacy into account. We regard this experiment model without noise processing as the basic model. The experiment analyzes their accuracies from two feature extraction operations PCA and CNN respectively. The experimental batch of all robots in one round is set to 600, and the initial model learning rate is set to 0.01. PCA establishes a mapping of 784 to 60. CNN structure designs two convolutional layers with convolution kernel sizes of 5 * 5 and 5 * 5 respectively, and its max pool structure is 2 * 2. The subsequent hidden layer is a 1000-dimension fully-connected structure. More fully-connected dimensions can better match the training data and reflect the relationship between their characteristics. The corresponding output of softmax classifier is 10 category labels.

For the differential federated learning model, the difference between this model and the basic model lays in the processing of each robot's gradient results of the randomly sampled data. The differential model perturbs the gradient with a certain noise distribution, and the disturbed gradients are gathered to the cloud. The overall privacy loss (ε, δ) can be computed from the noise level σ. We record the accuary for different (ε, δ) in Figure x. In our experiment, we set σ= [2, 4], and ε =2 and

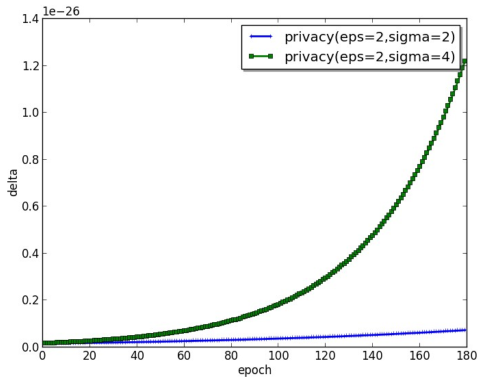

Fig. 7. The value of as a function of the training epoch using moments accountant.

compute the value of δ as a function of the training epoch. Figure 7 shows two curves corresponding to, respectively, σ= [2, 4] using the moments accountant. When epoch is 160 and the noisy parameter σ is 2, we get the (2, 0.8*e^{-26})-differential privacy. Relatively σ = 4, then (2, 6.27e^{-28}) -differential privacy.

In the case of the PCA structure, we compare the experiments with and without the noise processing. When there is no noise processing, the gradient information gathered by each robot remains as it is. However, for the differential federated learning model, we introduce the variable parameter σ, where its value is set to 2, and 4, respectively, and we calculate the trend of the accuracy with increasing epochs under corresponding parameters. Figure 8 shows the accuracy results under different noise parameters in the case of PCA structure, where the data in each test is 100 images randomly selected from the 10000 test images. In the plot, we show the evolution of the recognition accuracy as a function of the number of epochs as well as the corresponding σ (sigma) value, each epoch means 10x rounds. The accuracy is improved continuously with more knowledge integration as the epoch increasing, though the growing trend is slowing down. "Non-privacy" means that the gradient is not noise-processed. Among them, σ is the intensity of noise. The larger the value σ is, the less likely the model will be cracked, and of course its accuracy will be affected to some extent.

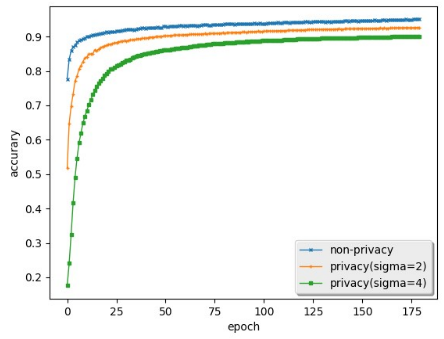

Fig. 8. The MNIST accuracies under different noise parameters in the case of PCA structure.

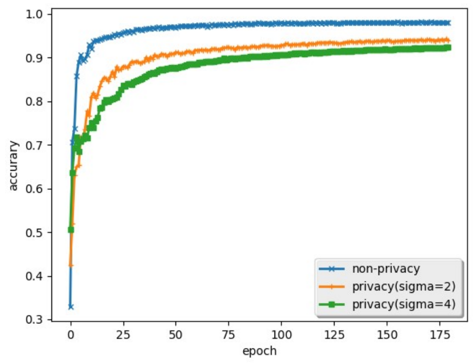

Fig. 9. The MNIST accuracies under different noise parameters in the case of CNN structure.

Meanwhile, for the CNN structure, we also conduct experiments similar to those under PCA. We also calculate the trend of the accuracy with increasing epochs under corresponding parameters. Figure 9 also shows the relationship between the accuracy and privacy protection with the value of σ. Compared with Figure 6, CNN has advantages in feature extraction, which can achieve higher accuracy faster than PCA structure. During the training process, the accuracy of CNN was 2% higher than the accuracy of PCA under the same epoch.

B. CIFAR-10

We also evaluate the experiments on the CIFAR-10 dataset. The CIFAR-10 dataset consists of 60000 32 * 32 color images in 10 classes such as airplane, bird, dog, truck, with 6000 images of each class, and is partitioned into 50000 training images and 10000 test images. In order to use the model structure mentioned earlier, we use the skimage [33] library to compress the image into the size of 28 * 28, and the RGB three-channel into one channel. We take 10,000 images from the dataset as public data, the remaining 40,000 samples are utilized in a disjoint manner, giving an average of 400 to each robot.

1) Recognition latency optimization. The latency optimization of CIFAR-10 is basically consistent with the experimental results of MNIST in terms of queuing latency and propagation latency. The main difference lays in the latency of the data processing of the edge robots and the processing latency of the request processing on the cloud, which is verified by the experimental results. The basic difference is less than 0.1ms or so, this slight difference mainly comes from the difference of the neural network structure.

2) Experiments of accuracy and differential privacy in federated learning model. TensorFlow is applied to re-construct a convolution model for CIFAR-10. Here, we do not use PCA structure any longer, for CIFAR-10 dataset is more complicated than MNIST and PCA cannot obtain its features efficiently. The network contains two convolutional layers, followed by two fully connected layers and a softmax classifier. The structure of each convolutional layer is the same, both 5 * 5 kernel, 2 * 2 max pools.

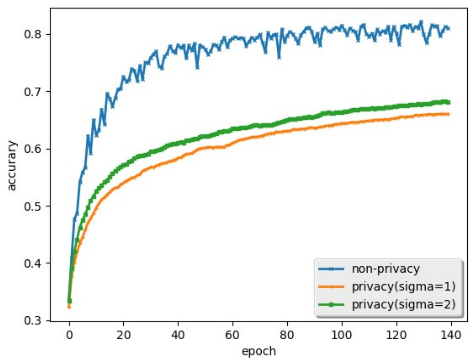

Fig. 10. The CIFAR-10 accuracies under different noise parameters in the case of CNN structure.

For the differential federated learning model, the difference between this model and the basic model lays in the processing of each robot's gradient results of the random sample data. The differential model perturbs the gradient with a certain noise distribution, and the disturbed gradients are gathered to the cloud. The overall privacy loss (ε, δ) can be computed from the noise level σ, which is set to 1 and 2 respectively. In Figure 9, we show the trend of accuracy under different σ. The variation of noise parameters has a significant impact on accuracy, which is different from the previous MNIST experiment. In the process of differential federated learning training, compared with the non-privacy model, the training accuracy of privacy training model has been dramatically reduced. The main reason for our analysis is that the sample characteristics of the CIFAR-10 training dataset are relatively complex and the model parameters are highly correlated. Utilizing the noise added to the training model, the gradient will seriously affect the stability of the model. Therefore, we will study this question in our future work.

VI. Conclusion

Nowadays, robots are typical ubiquitous devices that can generate a big amount of data and seriously call for the real-time data analysis. In this paper, we proposed a real-time data processing architecture for multi-robots based on the differential federated learning. On the cloud, we train a global model based on the multiple uploaded model parameter updates from distributed robots with differential privacy processing, and then the cloud would distribute the shared model to robots for local application in each round. Through theoretical analysis and experiments, our architecture can obtain the shared knowledge business values under the premise of ensuring real-time data processing and data privacy. In our future work, in addition to the robotic recognition work in the real scenario, we will apply our architecture into more real-time IoT applications.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under Grant 61202488 and 61751208.

References

[1] Gubbi J, Buyya R, Marusic S, et al. Internet of Things (IoT): A vision, architectural elements, and future directions[J]. Future Generation Computer Systems, 2013, 29(7): 1645-1660.

[2] Sundmaeker H, Guillemin P, Friess P, et al. Vision and challenges for realising the Internet of Things[J]. Cluster of European Research Projects on the Internet of Things, European Commision, 2010, 3(3): 34-36.

[3] Kehoe B, Patil S, Abbeel P, et al. A survey of research on cloud robotics and automation[J]. IEEE Transactions on automation science and engineering, 2015, 12(2): 398-409.

[4] Dastjerdi A V, Buyya R. Fog computing: Helping the Internet of Things realize its potential[J]. Computer, 2016, 49(8): 112-116.

[5] Cloudwards. [Online]. Available: https://www.cloudwards.net/review/zoolz/.

[6] Y. Li, H. Wang, B. Ding, P. Shi, and X. Liu. Toward QoS-Aware Cloud Robotic Applications: A Hybrid Architecture and Its Implementation[C]. IEEE Conference on UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld, 2016: 33-40.

[7] Shi W, Cao J, Zhang Q, et al. Edge computing: Vision and challenges[J]. IEEE Internet of Things Journal, 2016, 3(5): 637-646.

[8] Konečný J, McMahan H B, Yu F X, et al. Federated learning: Strategies for improving communication efficiency[J]. arXiv preprint arXiv:1610.05492, 2016.

[9] Cynthia Dwork, Aaron Roth. The algorithmic foundations of differential privacy. Foundations and Trends in Theoretical Computer Science[J]. 2014, 9(3-4):211-407.

[10] Collet A, Berenson D, Srinivasa S S, et al. Object recognition and full pose registration from a single image for robotic manipulation[C]. IEEE International Conference on Robotics and Automation, 2009: 48-55.

[11] Thrun S, Leonard J J. Simultaneous localization and mapping[M]. Springer handbook of robotics. Springer Berlin Heidelberg, 2008: 871-889.

[12] Bao H, Wang H M, Ding B, et al. Cloud-Based Knowledge Sharing in Cooperative Robot Tracking of Multiple Targets with Deep Neural Network[C]. International Conference on Neural Information Processing. Springer, Cham, 2017: 71-80.

[13] Duan Y, Yu X. Multi-robot system based on cloud platform[C]. Guidance, Navigation and Control Conference (CGNCC), 2016 IEEE Chinese. IEEE, 2016: 614-617.

[14] CloudSight. [Online]. Available: http://cloudsightapi.com/.

[15] Google Cloud Vision API. [Online]. Available: https://cloud.google.com/vision/.

[16] Image++. [Online]. Available: http://www.imageplusplus.com/.

[17] Schaffer A. Robots That Teach Each Other[J]. MIT Technology Review's 10 Breakthrough Technologies of 2016.

[18] Beetz M, Tenorth M, Winkler J. Open-EASE[C]. IEEE International Conference on Robotics and Automation (ICRA), 2015: 1983-1990.

[19] Luan T H, Gao L, Li Z, et al. Fog computing: Focusing on mobile users at the edge[J]. arXiv preprint arXiv:1502.01815, 2015.

[20] Zhang C, Bengio S, Hardt M, et al. Understanding deep learning requires rethinking generalization[J]. arXiv preprint arXiv:1611.03530, 2016.

[21] Papernot N, Abadi M, Erlingsson Ú, et al. Semi-supervised knowledge transfer for deep learning from private training data[J]. arXiv preprint arXiv:1610.05755, 2016.

[22] Holland S M. Principal components analysis (PCA)[J]. Department of Geology, University of Georgia, Athens, GA, 2008: 30602-2501.

[23] Su H, Chen H. Experiments on parallel training of deep neural network using model averaging[J]. arXiv preprint arXiv:1507.01239, 2015.

[24] C. Dwork. A firm foundation for private data analysis[J]. ACM, 2011, 54(1):86-95.

[25] C. Dwork, F. McSherry, K. Nissim, and A. Smith.Calibrating noise to sensitivity in private data analysis[J]. In TCC, Springer, 2006: 265-284.

[26] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation[J]. In EUROCRYPT, Springer, 2006: 486-503.

[27] Liu H H. Introduction to Queuing Theory[J]. Software Performance and Scalability: A Quantitative Approach, 135-176.

[28] Sun C, Shrivastava A, Singh S, et al. Revisiting unreasonable effectiveness of data in deep learning era[J]. arXiv preprint arXiv:1707.02968, 2017, 1.

[29] Abadi M, Chu A, Goodfellow I, et al. Deep learning with differential privacy[C]. ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016: 308-318.

[30] THE MNIST DATABASE of handwritten digits. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[31] CIFAR-10 and CIFAR-100 datasets. [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html.

[32] TensorFlow. [Online]. Available: https://github.com/tensorflow/tensorflow.

[33] skimage - skimage v0.14dev docs. [Online]. Available: http://scikit-image.org/docs/dev/api/skimage.html.

[34] Razzaque M A, Milojevic-Jevric M, Palade A, et al. Middleware for Internet of Things: A Survey[J]. IEEE Internet of Things Journal, 2016, 3(1):70-95.

[35] Neumann P, Montavont J, Noël T. Indoor deployment of low-power wide area networks (LPWAN): A LoRaWAN case study[C]. IEEE 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), 2016: 1-8.

[36] Geyer R C, Klein T, Nabi M. Differentially Private Federated Learning: A Client Level Perspective[J]. 2017.