Privacy-preserving Activity and Health Monitoring on Databox

Yuchen Zhao, Hamed Haddadi
UK Dementia Research Institute
Imperial College London
{yuchen.zhao19,h.haddadi}@imperial.ac.uk

Severin Skillman, Shirin Enshaeifar,
Payam Barnaghi
UK Dementia Research Institute
University of Surrey
{s.skillman,s.enshaeifar,p.barnaghi}@surrey.ac.uk

ABSTRACT

Activity recognition using deep learning and sensor data can help monitor activities and health conditions of people who need assistance in their daily lives. Deep Neural Network (DNN) models to infer the activities require data collected by in-home sensory devices. These data are often sent to a centralised cloud to be used for training the model. Centralising the data introduces privacy risks. The collected data contain sensitive information about the subjects. The cloud-based approach increases the risk that the data be stored and reused for other purposes without the owner's control. We propose a system that uses edge devices to implement activity and health monitoring locally and applies federated learning to facilitate the training process. The devices use the Databox platform to manage sensor data collected in people's homes, conduct activity recognition locally, and collaboratively train a DNN model without transferring the collected data into the cloud. We illustrate the applicability of the processing time of activity recognition on edge devices. We use a hierarchical model in which a global model is generated in the cloud, without requiring the raw data, and local models are trained on edge devices. The activity inference accuracy of the global model converges to a sufficient level after a few rounds of communication between edge devices and the cloud.

CCS CONCEPTS

• **Computer systems organization → Client-server architectures**; • **Computing methodologies → Supervised learning by classification**.

KEYWORDS

Edge Computing, Activity Recognition, Federated Learning

ACM Reference Format:
Yuchen Zhao, Hamed Haddadi, Severin Skillman, Shirin Enshaeifar,, and Payam Barnaghi. 2020. Privacy-preserving Activity and Health Monitoring on Databox. In *3rd International Workshop on Edge Systems, Analytics and Networking (EdgeSys '20), April 27, 2020, Heraklion, Greece.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3378679.3394529

EdgeSys '20, April 27, 2020, Heraklion, Greece

ACM ISBN 978-1-4503-7132-2/20/04…$15.00
https://doi.org/10.1145/3378679.3394529

1 INTRODUCTION

The rapid progress of Internet of Things (IoT) makes it possible to deploy a large number of low-cost devices that can continuously collect different types of data, from ambient environments such as room temperature and brightness to personal data such as basic physiological data. With the help of modern machine learning models such as Deep Neural Networks (DNNs), it is possible to recognise people's activities by analysing in-home monitoring data. Machine learning methods for activity recognition [8] enable us to monitor and predict people's health and activities to help those who need assistance and facilitate timely interventions to avoid preventable adverse health conditions [7].

The training and inference phases of activity recognition often require high computational resources which are usually provided on the cloud. Sending large volumes of raw data to the cloud poses privacy issues. The in-home sensory data are often personal, and users often do not want to share them beyond their controlled home environment. Ideally, for running deep learning on private data, we hope that we do not need to release the raw data for both the training and inference phases [18].

Edge computing [19] provides an alternative scheme to those systems that directly release data to the cloud. It deploys devices at the edge of the network, which is close to where data are generated (i.e., IoT sensors and devices in our scenarios), and brings computation to data. The computational resources on the devices make it possible to run many deep-learning applications, including activity recognition at the edge [6] without releasing raw data to the cloud, meanwhile providing lower latency and network traffic.

We propose a system that provides activity and health monitoring locally using in-home sensory data. In our system, an edge device uses the Databox platform [15] as a gateway to aggregate and manage the in-home sensory data. The machine learning method conducts both model training and activity inference processes locally. To facilitate the training process, we coordinate the edge devices, as clients, to form a federated learning (FL) system [14] with a cloud server. The server periodically sends a global DNN model to a selected set of clients and asks them to use their data to train the model locally. The selected clients then send the trained models back to the server, which aggregates these models into a global model for next training cycles. By this means, all clients collaboratively train a global model for activity recognition through repeatedly updating models with the cloud server, without releasing their raw data to either other clients or the server.

We evaluated our proposed system on a real edge device and also through simulation, on three popular activity recognition datasets. First, we implemented the local activity recognition using a Long Short-Term Memory (LSTM) model [12] on a Raspberry Pi 4 Model B

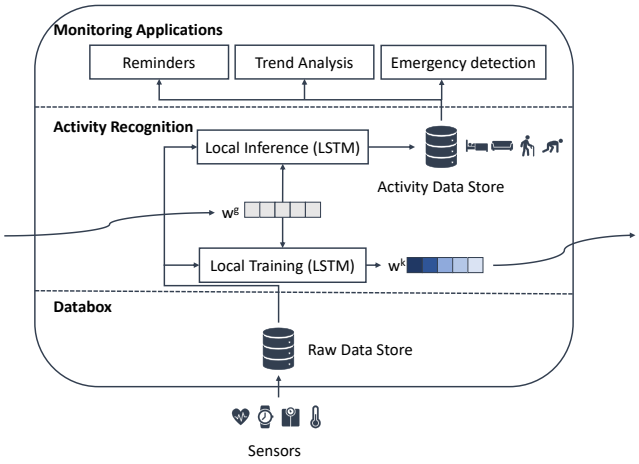

Figure 1: System structure inside an edge gateway using Databox as the underlying platform. The activity recognition layer receives global models (w^g), trains local models (w^k), and conducts inference locally. Inferred activities are used by health and activity monitoring applications.

and evaluated the processing time of activity recognition inference. We then simulated a FL scenario wherein 100 clients collaboratively train a global model, with the state-of-the-art training strategy [9] of LSTM. We have evaluated the inference accuracy of the system after each communication round. Our preliminary results show that the processing time of local activity inference on an edge device is feasible for low-cost device deployments. The proposed system can also achieve a stable level of inference accuracy after a few rounds of communication between clients and the cloud server.

2 SYSTEM DESIGN

The proposed system provides the following functionalities: (i) aggregation and management of in-home sensory data to support activity recognition; (ii) inferring people's activities locally on edge devices; and (iii) enabling different edge devices to collaboratively train a DNN model without releasing their raw data to the cloud. We now introduce how we achieve these goals through our design considerations.

2.1 Databox as an edge gateway

We build our system on top of the Databox platform [15], which provides functionalities for personal data management and access control. It can be deployed on edge devices such as Raspberry Pi and Intel NUC at people's homes as an edge gateway to receive sensor and IoT data, and aggregate these data for local applications such as activity recognition. In Databox, collected data are organised as *data stores* and can only be accessed through *drivers*, as shown in the bottom layer in Fig. 1. The default access control mechanism of Databox logs and manages operations of Databox Apps on data stores. To simplify the diagram, we do not show the driver and manager components of Databox in Fig. 1.

The reason for building our system on Databox is that the system locally converts in-home sensory data (i.e., raw data) to activity information, which can be used for health and activity monitoring. However, third parties may develop these applications. Thus their access to and operations on activity data, especially when requesting the activity data to leave people's homes, should be managed. Databox provides an efficient way to monitor and control local applications' behaviours.

2.2 Activity recognition at the edge

The key functionality of our system is to infer people's activities locally using collected data on Databox. Existing research [4] has shown that edge devices are capable of performing inferences locally using trained DNN models, which can reduce application latency. In addition, local inference does not need to send raw data to the cloud, which alleviate a major privacy concern that is commonplace in traditional cloud-based deep-learning applications.

We build an activity recognition layer on top of the Databox platform. It contains an LSTM network [12] and conducts two tasks, which are *local training* and *local inference* on Databox. The network has a sequence of LSTM cells, each of which has several hidden states. An LSTM cell takes time-series data (e.g., a sequence of sensor readings) as input and correspondingly outputs a series of hidden states. Thus the output of the network depends on its model parameters and input series. During the local training phase, our system feeds a series of local sensor data into the network, compares output states with actual activities (i.e., labelled data), and tunes the parameters of the network to minimise the errors in the output. The system can obtain labelled data by using publicly accessible data or asking users to conduct daily activities from time to time. For local inference, it receives a new LSTM network (i.e., the global model) from a cloud server. Through the data access mechanism of Databox, it retrieves the collected data within a small time window as current data and uses them as input to the global model. It uses the output of the model as the current activities of users and organises them as an *activity data store*, which can be used by local health and activity monitoring applications such as user reminders and emergency detection.

The activity recognition layer protects users' privacy by keeping the raw data locally for both the training and inference phases. It only sends the trained local model to the cloud (see Sec. 2.3), and the model can be further processed by tighter privacy guarantees such as differential privacy before leaving Databox. The inferred activities are also kept locally as a data store and used by local applications. The default mechanism of Databox controls local applications' access to these activities.

2.3 Model training through federated learning

There are two major challenges of training DNN models locally on an edge device. First, the training process needs typically to repeat many iterations, i.e., *epochs*, in order to optimise the parameters of a model and achieve high inference accuracy. Doing so on a single client consumes many computational resources. Second, a new client with insufficient local data may not be able to train an accurate model. We use federated learning (FL) [14] to address these issues and facilitate the training process in our system.

In the FL system, a cloud server communicates with clients and helps them train a global model w^g. As shown in Fig. 2, during

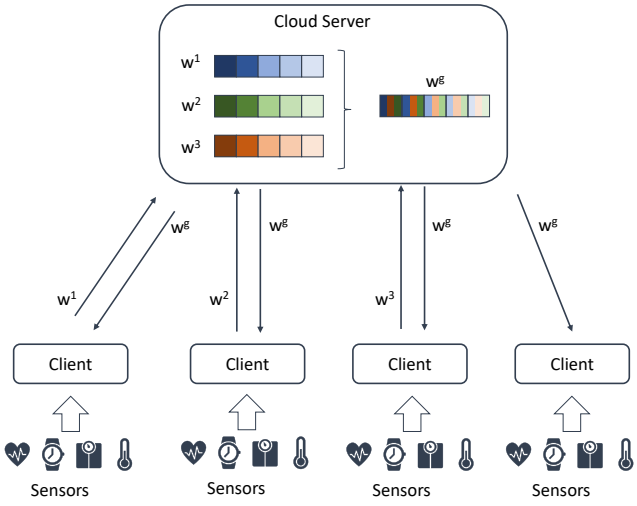

Figure 2: Federated learning structure. Each client sends its local model (w^k) to the cloud server. The cloud server generates a global model (w^g) from averaging the received local models and sends it back to all the clients for local inference.

Input K: number of all clients; C: fraction of clients;
1: initialise w_0^g at $t = 0$
2: **foreach** communication round t **do**
3: $S_t \leftarrow$ randomly selected $K \cdot C$ clients
4: **foreach** client $k \in S_t$ **do**
5: $w_{t+1}^k \leftarrow LocalTraining(k, w_t^g)$
6: **end for**
7: $w_{t+1}^g \leftarrow \sum_{k \in S_t} \frac{n_k}{n} w_{t+1}^k$
8: **end for**

Figure 3: FederatedAveraging (FedAvg) algorithm on the cloud server. n_k is the number of local samples of client k and n is the number of the samples of all selected clients.

each communication round the server randomly selects a fraction of clients (indexed by k) and sends the current w^g to them. Each selected client uses its local data to train the received w^g through a small number of epochs and produces a new local model w^k. All produced new local models are then sent back to the server. As the new local models have different parameter values, the server uses the *FederatedAveraging (FedAvg)* algorithm termed by McMahan et al. [14] and all the w^k to calculate a weighted average model as the new w^g for the next communication round. Fig. 3 shows the pseudo-code of the cloud server part of the FedAvg algorithm.

By repeating rounds of communication between the cloud server and clients, w^g is continuously updated with contributions from different clients. Each client conducts local training only when it is selected, and the training process only takes a small number of epochs. More importantly, the server only receives resulting models instead of raw data from clients.

3 EVALUATION

We have evaluated our system in two steps. First, we evaluated the processing time of local activity recognition by using a trained LSTM model. We used a Raspberry Pi 4 Model B for the edge device. We then evaluated the accuracy of the global model by simulating the FL system with repeated rounds of communication. We have used three activity recognition datasets related to household activity and health monitoring in our evaluation.

Our preliminary results show that: (i) the processing time of local inference on an edge device is acceptable; and (ii) the global model achieves a sufficient level of inference accuracy after a few rounds of communication between clients and the cloud server.

3.1 Datasets

The time-series activity datasets used in our evaluation include the Opportunity (Opp) dataset [5], the Daphnet Freezing of Gait (DG) dataset [2], and the PAMAP2 dataset [17]. The Opp dataset contains kitchen activities collected from 4 participants. The DG dataset contains incidents of freezing of gaits, which is an important contributor to falls in patients with Parkinson's Disease, collected from 10 participants. The activities and incidents in both datasets are short-term and non-repeated. The PAMAP2 dataset contains long-term and repeated household and exercise activities collected from 9 participants and is used to evaluate our system's inference accuracy on long-duration activities. We follow the same data preprocessing procedure in work by Hammerla et al. [10] to make training and testing datasets. Table 1 shows the number of input features, number of output classes, and the sizes of training and testing datasets generated from different raw datasets.

Table 1: Characteristics of the 3 activity datasets in our experiments.

Dataset	Activities	Features	Classes	Training	Testing
Opp	Kitchen	79	18	651k	119k
DG	Gait	9	3	792k	81k
PAMAP2	Household & Exercise	52	12	473k	83k

In our simulated FL system, for each raw training dataset, we generated several local training datasets and allocated them to 100 clients. The number of clients is more than the number of participants in each dataset, which means that the clients' data are subsets of participants' data. This allows us to simulate situations where data are sparse and not continuously recorded. To allocate n_k samples to a client, we use two strategies to generate local training datasets from raw training datasets.

IID local training datasets. In this strategy, we split a raw training dataset into 100 divisions. The samples for each client uniformly distribute among these divisions. In each division, we randomly locate a small time window with the length of $\frac{n_k}{100}$ and retrieve the samples within the time window as the samples of the division. Samples of all divisions are concatenated as the local training dataset for a client. By this means, we can make sure that the local training datasets of different clients have similar distributions.

Non-IID local training datasets. We assume that in real-world FL, each client's local training dataset could be skewed. In this setting, we randomly locate a time window with a length of n_k among the entire raw training dataset and allocate all the samples within the time window to a client. Thus, each client's local training dataset can only cover a continuous region of the raw training dataset. We want to evaluate how this skewed distribution of local training datasets affects the inference accuracy of the global model.

3.2 Setup

To evaluate the processing time of local inference, we train LSTM models (1 LSTM cell, 256 hidden states, learning rate 0.01, 1 fully-connected layer) using PyTorch on the entire training datasets. We test the trained models locally on a Raspberry Pi 4 Model B, which is a typical model of the edge device. We use a one-second time window to sample the entire testing datasets and measure the local inference time on each sampled subset. The reason of using a one-second time window is to evaluate the processing time within a unit time period. In practical conditions of activity and health monitoring, the local inference happens continuously. Table 2 shows the system specifications of the Raspberry Pi used in our evaluation.

Table 2: System specs of our used Raspberry Pi 4 Model B.

CPU	Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
RAM	4GB LPDDR4-3200 SDRAM
Storage	SanDisk Ultra 32GB microSDHC Memory Card
OS	Ubuntu Server 19.10

To evaluate the inference accuracy of the global model, we simulated a vanilla FL system. The cloud server starts with a global LSTM model (same network structure as the one in local inference) with randomised initial parameters. In each communication round, the server randomly selects 10% of 100 clients and sends the global model to them. Each selected client uses its local training data to update the parameters of the global model through 2 local epochs. For each epoch, we follow the LSTM training procedure proposed by Guan and Plötz [9], which uses bagging (bootstrap aggregating) strategies by randomising batch sizes and sequence lengths. The client uses cross-entropy loss as the loss function and Adam optimisation during the training phase and generates a local model with updated parameters. At the end of each round, all local models are sent back to the server. The local models are used to generate a new global model. We evaluate the inference accuracy (i.e., the ratio of correctly inferred samples among all tested samples) of the resulting global model against the whole test datasets. The communication processes (100 rounds on Opp and DG, 200 rounds on PAMAP2) are repeated until the inference accuracy reaches a stable level. For each raw dataset, we run 50 replicates of simulations and use the average value of the replicates as results.

We keep the network structure of the LSTM models and the local training process simple (i.e., only 1 LSTM cell and 2 local epochs). The reason is that we want to reduce the consumption of computational resources on edge devices. We want to know how much FL can boost the training phase when both model complexity (1 LSTM cell) and individual consumption (2 local epochs) are low.

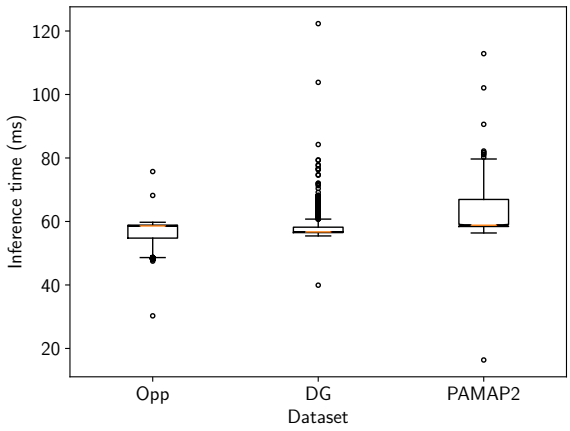

Figure 4: Inference time on a Raspberry Pi 4 Model B. Each data point represents the processing time (ms) of the local LSTM model to infer the activities of the samples within a one-second time window (33 samples).

3.3 Inference time

Fig. 4 shows the distributions of the processing time of local inference on samples within a one-second time window on different datasets. As the sample rate of the datasets is 33Hz, each time window contains 33 samples. For all the three datasets, the average inference time is lower than 70ms (Opp: 56 ± 4ms, DG: 59 ± 4ms, PAMAP2: 62 ± 5ms). Our results show that for one second of time-series sensor readings, the overhead of local inference on them is lower than 7%, which is acceptable. Although this evaluation does not take the data transmission time into account and the dimensionality is not as high as those in image or video-based activity recognition, we believe that this preliminary result shows the potential of local activity inference on edge devices.

3.4 Inference accuracy

We first look at the results with IID training data. Fig. 5 shows the inference accuracy of our system during each communication round on 3 datasets. In the early rounds of the simulation, the initial accuracy depends on the evaluated dataset. As the rounds of communication increase, the accuracy goes up and converges at a stable level on all 3 datasets. The speed of the convergence also differs in different datasets. For example, the initial accuracy on Opp is close to its stable accuracy and converges gradually. The initial accuracy on PAMAP2 is much lower than its stable accuracy but converges to the stable level after about 25 rounds. The initial accuracy on DG is almost the same as its stable accuracy. As DG has much fewer activity classes (3) than PAMAP2 (12) and Opp (18), we speculate that the number of activity classes may affect the speed of convergence in our system.

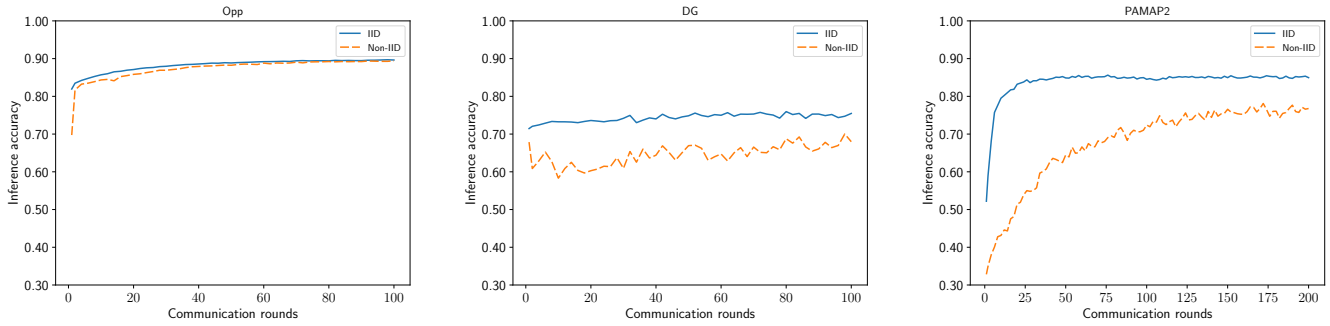

Figure 5: Inference accuracy on different datasets (100 communication rounds on Opp and DG, 200 communication rounds on PAMAP2) with IID and Non-IID data. Accuracy with IID data is higher and converges faster than that with Non-IID data.

The inference accuracy with Non-IID data shows similar trends as that with IID data does but has a few differences. As expected, the initial accuracy on Non-IID data is lower than that with IID data. This is because that the Non-IID data of each client only covers a small continuous region of the training sets. Thus the models trained on these local data are skewed. As the rounds of communication increase, in some datasets, this kind of skewness is averaged by the FedAvg algorithm. For example, on the Opp dataset, the Non-IID inference accuracy goes up rapidly and becomes close to the IID inference accuracy. On the DG dataset, the Non-IID accuracy reaches a stable level in the very beginning, but has more fluctuation and is constantly lower than the IID accuracy. The accuracy with Non-IID data in PAMAP2 is also steadily lower than that with IID data and converges more slowly.

Our experimental results demonstrate promising inference accuracy with IID data, which is an ideal situation of our system. When the local data are not skewed, the accuracy of the trained global model can converge to a stable level within a few communication rounds. The accuracy of datasets that have fewer activity classes may converge faster. The accuracy with Non-IID data is affected by the skewed local data and is lower than that with IID data in general. This can be considered as a non-ideal situation of our system, wherein each client only has data for specific times of the day or specific types of activity. However, the difference between the accuracies with IID and Non-IID data depends on the data, i.e., activities, which would be further investigated in our future research.

4 DISCUSSION

Our system aims to provide health and activity monitoring based on local inference, and meanwhile protect users' privacy by keeping their raw data stored on the Databox. The only data that are sent to the cloud are locally trained models. By this means, our system provides a direct privacy guarantee against an "honest but curious" cloud server, which follows the service protocol between it and its users but may store and reuse the received data. Adversaries with more advanced abilities, however, may still be able to breach users' privacy through analysing the released models. For example, membership inference attacks [21] and model inversion attacks [11] may reveal information in the original training datasets. These threats are introduced by malicious users, which were not taken into account in this paper, and may affect the expected privacy level of our system. To address these possible issues, privacy-enhancing technologies such as differentially private deep learning [1], either on the clients or on the cloud, are needed in order to provide a higher level of privacy. Inevitably, it may cause additional expense on utility (e.g., inference accuracy) and cost (e.g., computational resources). The trade-offs between them need to be further investigated.

Since federated learning is an open structure, it is necessary to control the quality of the contributions from clients. The quality of data on different clients may be different due to sensors' connectivity and users' daily activity patterns. As a consequence, the contributions from their local training processes are unlikely the same. In order to promote the contributions from clients that have more useful data than others, a reputation system or a trust-based system can be introduced. Such a system can prioritise the contributions from a subset of users when calculating the global model and make the inference accuracy converge faster. We plan to explore and evaluate these systems in the future.

5 RELATED WORK

Edge computing has facilitated many novel deep learning applications at the edge, such as computer vision, natural language processing, and IoT based human activity recognition. For instance, one application is fall detection from accelerometer readings [3, 16], which is useful in healthcare monitoring. Uddin [22] also proposes a smart healthcare system at the edge using electrocardiography (ECG), magnetometer, accelerometer, and gyroscope readings to recognise human activities. The experimental results show that a Recurrent Neural Network (RNN) trained on these data can detect activities with high accuracy at the edge. Apart from sensor readings, video data can also be used for activity recognition. For example, the EdgeEye system proposed by Liu et al. [13] can provide API to applications in the local network for real-time video analytics. To help realise these activity recognition systems at the edge, Zhang et al. [24] propose a general framework, including architecture, algorithms, and implementation. These existing systems and frameworks, however, rely on a cloud server that performs both the training and inference phases on users' raw data. In this paper,

we focus on addressing the privacy issues, which is an important aspect of activity recognition at the edge, with the help of local activity inference on edge devices and federated learning.

Edge devices are one of the solutions that can control the disclosure of local data to the cloud [23]. Platforms such as Databox [15] can be deployed on edge devices to collect and manage sensor and IoT data, on top of which we can implement activity recognition locally. A cluster of such devices can cooperate in the training phase by sharing models instead of raw data with each other. For example, Shokri and Shmatikov [20] propose a distributed system that allows multiple parties to selectively send the gradients of their local models to the cloud, thereby jointly learning a model together without disclosing the raw data. Similarly, the federate learning paradigm [14] allows multiple parties to send their models to a cloud server that uses the FedAvg algorithm to generate a global model, which can be used by all the parties. It not only protects the privacy of individuals but also boosts the training process.

6 CONCLUSIONS

Health and activity monitoring using sensor and IoT data can greatly improve people's living conditions in their daily lives. However, privacy is one of the major issues in traditional systems where all the data are sent to a cloud server. In this paper, we propose a system that utilises the recent progress of edge computing and federated learning. Our proposed system uses the Databox platform to collect and manage sensor and IoT data and uses these data to infer activities locally on an edge device, based on which health and activity monitoring can be realised. It uses a cloud server to coordinate different devices to jointly train a global LSTM model without releasing their raw data, which provides a direct protection on users' privacy. Our experimental results show that the processing time of local inference on an edge device is acceptable. Meanwhile, the inference accuracy of our system can converge to a sufficient and stable level after a few rounds of communication between the clients and the server. As an ongoing work, we plan to introduce formal privacy enhancing technologies such as differential privacy into our system and evaluate the trade-offs between utility, privacy, and cost. We also plan to use reputation-based or trust-based mechanisms to optimise the contributions from individual devices, thereby increasing the inference accuracy of the system.

ACKNOWLEDGMENTS

This work was supported by the UK Dementia Research Institute.

REFERENCES

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep Learning with Differential Privacy. In *Proc. of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM Press, 308–318. https://doi.org/10.1145/2976749.2978318

[2] Marc Bachlin, Daniel Roggen, Gerhard Troster, Meir Plotnik, Noit Inbar, Inbal Meidan, Talia Herman, Marina Brozgol, Eliya Shaviv, Nir Giladi, and Jeffrey M. Hausdorff. 2009. Potentials of Enhanced Context Awareness in Wearable Assistants for Parkinson's Disease Patients with the Freezing of Gait Syndrome. In *Proc. of the 2009 International Symposium on Wearable Computers*. IEEE, 123–130. https://doi.org/10.1109/ISWC.2009.14

[3] Yu Cao, Peng Hou, Donald Brown, Jie Wang, and Songqing Chen. 2015. Distributed Analytics and Edge Intelligence. In *Proc. of the 2015 Workshop on Mobile Big Data*. ACM Press, 43–48. https://doi.org/10.1145/2757384.2757398

[4] Alejandro Cartas, Martin Kocour, Aravindh Raman, Ilias Leontiadis, Jordi Luque, Nishanth Sastry, Jose Nuñez-Martinez, Diego Perino, and Carlos Segura. 2019. A Reality Check on Inference at Mobile Networks Edge. In *Proc. of the 2019 International Workshop on Edge Systems, Analytics and Networking*. ACM Press, 54–59. https://doi.org/10.1145/3301418.3313946

[5] Ricardo Chavarriaga, Hesam Sagha, Alberto Calatroni, Sundara Tejaswi Digumarti, Gerhard Tröster, José Del R. Millán, and Daniel Roggen. 2013. The Opportunity Challenge. *Pattern Recognition Letters* 34, 15 (nov 2013), 2033–2042. https://doi.org/10.1016/j.patrec.2012.12.014

[6] Jiasi Chen and Xukan Ran. 2019. Deep Learning With Edge Computing: A Review. *Proc. IEEE* 107, 8 (aug 2019), 1655–1674. https://doi.org/10.1109/JPROC.2019.2921977

[7] Shirin Enshaeifar, Payam Barnaghi, Severin Skillman, Andreas Markides, Tarek Elsaleh, Sahr Thomas Acton, Ramin Nilforooshan, and Helen Rostill. 2018. The Internet of Things for Dementia Care. *IEEE Internet Computing* 22, 1 (jan 2018), 8–17. https://doi.org/10.1109/MIC.2018.112102418

[8] Shirin Enshaeifar, Ahmed Zoha, Andreas Markides, Severin Skillman, Sahr Thomas Acton, Tarek Elsaleh, Masoud Hassanpour, Alireza Ahrabian, Mark Kenny, Stuart Klein, Helen Rostill, Ramin Nilforooshan, and Payam Barnaghi. 2018. Health Management and Pattern Analysis of Daily Living Activities of People with Dementia Using In-home Sensors and Machine Learning Techniques. *PLOS ONE* 13, 5 (may 2018). https://doi.org/10.1371/journal.pone.0195605

[9] Yu Guan and Thomas Plötz. 2017. Ensembles of Deep LSTM Learners for Activity Recognition using Wearables. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 2 (jun 2017), 1–28. https://doi.org/10.1145/3090076

[10] Nils Y. Hammerla, Shane Halloran, and Thomas Plötz. 2016. Deep, Convolutional, and Recurrent Models for Human Activity Recognition using Wearables. In *Proc. of the 2016 International Joint Conference on Artificial Intelligence*. 1533–1540.

[11] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. 2017. Deep Models Under the GAN. In *Proc. of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM Press, 603–618. https://doi.org/10.1145/3133956.3134012

[12] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[13] Peng Liu, Bozhao Qi, and Suman Banerjee. 2018. EdgeEye. In *Proc. of the 2018 International Workshop on Edge Systems, Analytics and Networking*. ACM Press, 1–6. https://doi.org/10.1145/3213344.3213345

[14] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proc. of the 2017 International Conference on Artificial Intelligence and Statistics*. 1273–1282.

[15] Richard Mortier, Jianxin Zhao, Jon Crowcroft, Liang Wang, Qi Li, Hamed Haddadi, Yousef Amar, Andy Crabtree, James Colley, Tom Lodge, Anthony Brown, Derek McAuley, and Chris Greenhalgh. 2016. Personal Data Management with the Databox. In *Proc. of the 2016 ACM Workshop on Cloud-Assisted Networking*. 49–54. https://doi.org/10.1145/3010079.3010082

[16] J. Pena Queralta, T. N. Gia, H. Tenhunen, and T. Westerlund. 2019. Edge-AI in LoRa-based Health Monitoring. In *Proc. of the 2019 International Conference on Telecommunications and Signal Processing*. IEEE, 601–604. https://doi.org/10.1109/TSP.2019.8768883

[17] Attila Reiss and Didier Stricker. 2012. Introducing a New Benchmarked Dataset for Activity Monitoring. In *Proc. of the 2012 International Symposium on Wearable Computers*. IEEE, 108–109. https://doi.org/10.1109/ISWC.2012.13

[18] M. Sadegh Riazi, Bita Darvish Rouani, and Farinaz Koushanfar. 2019. Deep Learning on Private Data. *IEEE Security & Privacy* 17, 6 (nov 2019), 54–63. https://doi.org/10.1109/MSEC.2019.2935666

[19] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. 2016. Edge Computing: Vision and Challenges. *IEEE Internet of Things Journal* 3, 5 (oct 2016), 637–646. https://doi.org/10.1109/JIOT.2016.2579198

[20] Reza Shokri and Vitaly Shmatikov. 2015. Privacy-Preserving Deep Learning. In *Proc. of the 2015 ACM SIGSAC Conference on Computer and Communications Security*. ACM Press, 1310–1321. https://doi.org/10.1145/2810103.2813687

[21] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership Inference Attacks Against Machine Learning Models. In *Proc. of the 2017 IEEE Symposium on Security and Privacy*. IEEE, 3–18. https://doi.org/10.1109/SP.2017.41

[22] Md Zia Uddin. 2019. A Wearable Sensor-based Activity Prediction System to Facilitate Edge Computing in Smart Healthcare System. *J. Parallel and Distrib. Comput.* 123 (jan 2019), 46–53. https://doi.org/10.1016/j.jpdc.2018.08.010

[23] Shanhe Yi, Zhengrui Qin, and Qun Li. 2015. Security and Privacy Issues of Fog Computing: A Survey. In *Proc. of the 2015 International Conference on Wireless Algorithms, Systems, and Applications*. 685–695. https://doi.org/10.1007/978-3-319-21837-3_67

[24] Shaojun Zhang, Wei Li, Yongwei Wu, Paul Watson, and Albert Zomaya. 2018. Enabling Edge Intelligence for Activity Recognition in Smart Homes. In *Proc. of the 2018 IEEE International Conference on Mobile Ad Hoc and Sensor Systems*. IEEE, 228–236. https://doi.org/10.1109/MASS.2018.00044