Challenges of Privacy-Preserving Machine Learning in IoT

Mengyao Zheng[*]
Nanyang Technological University

Dixing Xu[*]
Nanyang Technological University

Linshan Jiang
Nanyang Technological University

Chaojie Gu
Nanyang Technological University

Rui Tan
Nanyang Technological University

Peng Cheng
Zhejiang University

ABSTRACT

The Internet of Things (IoT) will be a main data generation infrastructure for achieving better system intelligence. However, the extensive data collection and processing in IoT also engender various privacy concerns. This paper provides a taxonomy of the existing privacy-preserving machine learning approaches developed in the context of cloud computing and discusses the challenges of applying them in the context of IoT. Moreover, we present a privacy-preserving inference approach that runs a lightweight neural network at IoT objects to obfuscate the data before transmission and a deep neural network in the cloud to classify the obfuscated data. Evaluation based on the MNIST dataset shows satisfactory performance.

CCS CONCEPTS

• **Security and privacy** → **Domain-specific security and privacy architectures**; • **Computer systems organization** → **Sensor networks**.

KEYWORDS

Internet of Things, machine learning, privacy

ACM Reference Format:
Mengyao Zheng, Dixing Xu, Linshan Jiang, Chaojie Gu, Rui Tan, and Peng Cheng. 2019. Challenges of Privacy-Preserving Machine Learning in IoT. In *First International Workshop on Challenges in Artificial Intelligence and Machine Learning (AIChallengeIoT'19), November 10–13, 2019, New York, NY, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3363347.3363357

1 INTRODUCTION

With the advances of sensing and communication technologies, the Internet of Things (IoT) will become a main data generation infrastructure in the future. The drastically increasing amount of data generated by IoT will create unprecedented opportunities for various novel applications powered by machine learning (ML). However, various system challenges need to be addressed to implement the envisaged intelligent IoT.

IoT in nature is a distributed system consisting of heterogeneous nodes with distinct sensing, computation, and communication capabilities. Specifically, it consists of massive mote-class sensors deeply embedded in the physical world, personal devices that move with people, widespread network edge devices such as wireless access points, as well as the cloud backend. Implementing the fabric of IoT and ML faces the following two key challenges:

- **Separation of data sources and ML computation power:** Most IoT data will be generated by the end devices that often have limited computation resources, while the computation power needed by ML model training and execution will be located at the edge devices and in the cloud. Besides, the communication channels between the end devices and the edge/cloud are often constrained, in that they are limited in bandwidth, intermittent, and of long delays.
- **Privacy preservation:** As the end devices can be deeply embedded in people's private space and time, the data generated by them will contain privacy-sensitive information. To gain wide acceptance, the IoT-ML fabric must respect the human users' privacy. The lack of privacy preservation may even go against the recent legislation such as the General Data Protection Regulation in European Union. However, privacy preservation often presents substantial challenges to the system design.

Privacy-preserving ML has received extensive research in the context of cloud computing. Thus, it is of great interest to investigate whether the existing solutions can be applied in the context of IoT. To this end, this paper provides a taxonomy of the existing privacy-preserving ML approaches, which are classified into two categories: *privacy-preserving training* and *privacy-preserving inference*. The former, which has received considerable research attention, is comprised of *parameter transmission-based* and *data transmission-based* approaches. Each of them can be further divided into multiple sub-categories. In contrast, our literature survey shows that less research work concentrates on privacy-preserving inference. Since the computation and communication overheads are the key considerations in the design of IoT systems, we tentatively label the existing privacy-preserving ML approaches with *high-overhead*, *medium-overhead*, and *low-overhead* regarding computation complexity and *iterative communication*, *data swelling*, and *data retention/compression* regarding communication overhead. ML for IoT should be of low-overhead and data retention/compression.

From our survey, a number of privacy-preserving training approaches with various privacy protection objectives [10, 14, 21, 22, 26, 28, 29, 31, 33, 34, 39, 41, 45] can be labeled low-overhead and data

[*]The first two authors contributed equally to this research. This work was completed when they were with Xi'an Jiaotong-Liverpool University and visiting Nanyang Technological University.

AIChallengeIoT'19, November 10–13, 2019, New York, NY, USA
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-7013-4/19/11…$15.00
https://doi.org/10.1145/3363347.3363357

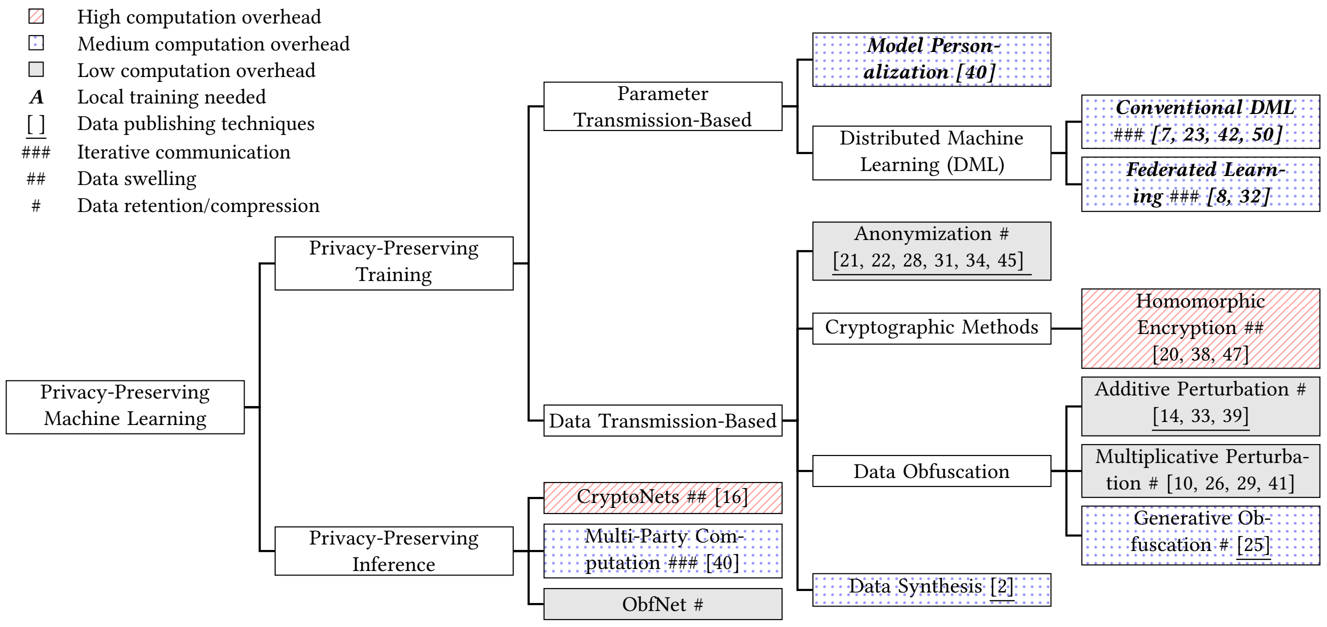

High computation overhead
Medium computation overhead
Low computation overhead
A Local training needed
[ ] Data publishing techniques
### Iterative communication
## Data swelling
# Data retention/compression
Privacy-Preserving Machine Learning
Privacy-Preserving Training
Privacy-Preserving Inference
Parameter Transmission-Based
Data Transmission-Based
Model Personalization [40]
Distributed Machine Learning (DML)
Conventional DML ### [7, 23, 42, 50]
Federated Learning ### [8, 32]
Anonymization # [21, 22, 28, 31, 34, 45]
Cryptographic Methods
Homomorphic Encryption ## [20, 38, 47]
Data Obfuscation
Additive Perturbation # [14, 33, 39]
Multiplicative Perturbation # [10, 26, 29, 41]
Generative Obfuscation # [25]
Data Synthesis [2]
CryptoNets ## [16]
Multi-Party Computation ### [40]
ObfNet #

Figure 1: The hierachical taxonomy of privacy-preserving machine learning approaches.

retention/compression. In contrast, the existing privacy-preserving inference approaches have high computation or communication overheads. Thus, we think that privacy-preserving inference for IoT should receive more research attention, since many IoT applications leverage pre-trained ML models for inference instead of training models from scratch. Thus, in the second part of this paper, we present a lightweight and voluntary privacy-preserving inference approach that is suitable for resource-constrained IoT objects. The approach is as follows. Given a well trained deep model that will be executed by the edge/cloud for inference, the approach trains a few dense layers such that the concatenation of these layers and the given deep model still yields satisfactory inference accuracy. At run time, an IoT end device executes these dense layers to obfuscate the inference data sample and sends the result to the edge/cloud for inference. Our approach is voluntary in that the deep model at the edge/cloud admits both original and obfuscated data samples. Each IoT end device in the system can choose to execute the dense layers for obfuscation or simply send the original data for inference. This design accommodates the end devices that cannot perform the obfuscation due to say limited computing capability. The evaluation based on the MNIST dataset [27] of handwritten digits shows that our obfuscation approach well protects the confidentiality of the raw form of the data samples and maintains the inference accuracy.

The remainder of this paper is organized as follows. §2 presents the taxonomy of the existing privacy-preserving ML approaches and their limitations in the context of IoT. §3 presents our lightweight and voluntary privacy-preserving inference approach and evaluation result. §4 concludes this paper and discusses future work.

2 EXISTING APPROACHES & LIMITATIONS

Fig. 1 illustrates the taxonomy of the existing privacy-preserving ML schemes. The nodes in a privacy-preserving ML system often have two roles: *participant* and *coordinator*. The participants are often the data generators (e.g., smartphones), whereas the coordinator (e.g., a cloud server) orchestrates the ML process. Since ML has two phases, i.e., training and inference, we classify the existing approaches into two groups at the top level. The *privacy-preserving training* schemes (§2.1) aim to learn a global ML model or multiple local ML models from disjoint local datasets which, if aggregated, would provide more useful/precise knowledge. Thus, the primary objective of privacy protection is to preserve the privacy of the data used for building an ML model in the training phase. Differently, *privacy-preserving inference* schemes (§2.2) focus on the scenario where a global ML model at the coordinator has been trained and the participants transmit the unlabeled data to the coordinator for inference. The aim is to protect the privacy of the input data in the inference phase and maintain the inference accuracy.

Privacy-preserving training schemes can be further classified based on whether the privacy-sensitive training data samples are transmitted or only the model parameters are transmitted for model training. Parameter transmission-based approaches (§2.1.1) include distributed machine learning and model personalization. The approaches that need to transmit local data samples (§2.1.2) can be classified into *anonymization*, *cryptographic methods*, *obfuscation*, and *data synthesis* according to the processing made on the training data. (1) Anonymization approaches de-identify data records but do not change the data of interest for model training. (2) Cryptographic methods apply cryptographic primitives to encrypt the data transmitted. (3) Obfuscation methods transform the training data vectors through additive perturbation, multiplicative perturbation, and generative obfuscation. (4) Data synthesis generates a new dataset that resembles the original dataset. Note that some of the data transmission-based approaches for privacy-preserving training are *data publishing* techniques, which focus on the proper sharing of the data or query results and in general do not explicitly address the problem of ML model training. Our taxonomy includes them for the completeness of the related work review.

Most existing privacy-preserving ML approaches were designed in the context of cloud computing. The participants are often resource-rich nodes from smartphones to cloud servers. In particular, the overhead of communications is not a key concern due to the availability of high-speed connections (e.g., wireline networks and 4G cellular networks). Differently, in the context of IoT, the participants are often resource-constrained devices. Moreover, the communication links among them are generally constrained. Therefore, in the review of the privacy-preserving training (§2.1) and inference (§2.2) approaches, we will qualitatively discuss their computation overhead and communication overhead. Here are our qualitative labels of the computation and communication overheads:

- Computation overhead: We classify the level of computation overhead into *high*, *medium*, and *low*, with homomorphic encryption, neural network training/inference, and additive/multiplicative noisification as the representative examples, respectively. The high-overhead computation tasks are in general infeasible for IoT end devices. For the medium-overhead computation, IoT end devices are increasingly capable of neural network inference computation due to the emerging inference chips such as Google's Edge TPU [19]. However, neural network training is still largely infeasible for IoT end devices at present.
- Communication overhead: We classify the communication overheads of the existing approaches into three categories: *iterative communication*, *data swelling*, *data retention/compression*, with distributed machine learning, homomorphic encryption, and additive/multiplicative perturbation as the representative examples, respectively. Specifically, distributed machine learning requires iterative model parameter exchanges among the training participants. Such iterative communications will cause significant challenges for IoT networks due to the bandwidth-limited and intermittent communication links. The ciphertexts produced by homomorphic encryption algorithms often have higher data volumes than the plaintexts. In contrast, the additive and multiplicative perturbation will retain and even reduce the data volumes.

Storage overhead is also a factor of concern for IoT end devices. However, this paper primarily focuses on computation and communication overhead due to the page limit.

2.1 Privacy-Preserving Training

The latest privacy-preserving training approaches that leverage distributed privacy-sensitive data to construct a global ML model or multiple local ML models can be divided into parameter transmission-based (§2.1.1) and data transmission-based (§2.1.2) techniques.

2.1.1 Parameter Transmission-Based Approaches. Approaches of this category transmit model parameters instead of data samples for model training. In this way, parameter transmission-based approaches to privacy-preserving training push computation towards participants rather than the coordinator.

Distributed Machine Learning. Distributed Machine Learning (DML) is a representative approach of this category. In DML [7, 8, 11, 23, 32, 42, 50], data owners do not reveal their own datasets to anyone in the training phase. In each iteration, the participants upload merely the locally computed parameters or gradients to the coordinator to achieve *collaborative learning* [43]. Conventional DML algorithms [7, 23, 42, 50] exploit the fact that the optimization algorithms based on stochastic gradient descent (SGD) can be parallelized, if the data held by different participants are independently and identically distributed (i.i.d.). Variants of SGD such as Selective SGD [42], parallel SGD [50], Alternating Direction Method of Multipliers SGD [7] and Downpour SGD [11] are normally used to update the model weights in the distributed fashion.

Federated learning [8, 32] is another prevailing DML approach with more generalized assumptions. In each iteration, a fraction of participants are randomly selected to retrain the model with their local data using the current global model as the starting point and then individually upload the local stochastic gradient descents. The coordinator will then average the gradient descents and update the global model. However, while federated learning [8, 32] manages to reduce communication overhead, it increases the local computation overhead.

Arguably, model parameters contain some information about the local training data. Therefore, in [8, 23, 42], differential privacy [13] is achieved by adding noises to the locally computed parameter updates. Such schemes thwart definitive inferences about an individual participant if an adversary intentionally collects the obfuscated model updates. Besides, secure data aggregation is applied to aggregate the updates from individual participants [6]. Phong et al. [37] also use additively homomorphic encryption [12] to encrypt model parameters in the federated learning scheme to prevent information leakage.

Limitations: First, training a deep model locally may be impractical for resource-constrained IoT devices. Second, many iterations are required for the learning process to converge [26], which results in substantial communication overhead. Due to the computation and communication overhead incurred, DML is mainly deployed in the context of cloud computing with enterprise settings. As shown in [3, 5], federated learning is vulnerable to backdoor attacks. Hitaj et al. [24] devise a powerful attack based on generative adversarial networks [18] for local data recovery. The attack is still effective even when the communicated parameters are perturbed for differential privacy and secure multi-party computation (MPC) [17] is applied.

Model Personalization. The aim of model personalization [40] is not to learn a global model from privacy-sensitive data owned by the participants, but to learn a personal model for each participant based on a public model trained with public data as the starting point. Specifically, the public model is firstly trained with a set of public data at the coordinator and then distributed to each participant. Then, each participant retrains the model with local data. The idea of transfer leaning [35] is leveraged to achieve better performance than the model training with merely local data.

Limitations: This approach may be ineffective for some tasks where the local classes have significantly different patterns from the public data. Additionally, the local training is unsuitable for resource-constrained IoT devices.

2.1.2 *Data Transmission-Based Approaches.*

This category of approaches allows participants to send local data samples to the honest-but-curious coordinator, while protects certain aspect or attribute of the data samples, e.g., user identity, data contents, or raw form of data. It has the following sub-categories: anonymization, cryptographic methods, data obfuscation, and data synthesis.

Anonymization. Anonymization techniques are designed to anonymize the participant's identity in a group of users, changing the value of quasi-identifiers and removing explicit identifiers. Since the aim is to remove the association between data entries and the data owner, the data samples of interest used for model training remains unchanged. For field-structured data, anonymization techniques include k-anonymity [45], l-diversity [31], and t-closeness [28]. The k-same family of algorithms [21, 22, 34] are designed to de-identify face images.

Limitations: As analyzed in [31], anonymization techniques are vulnerable to *homogeneity attack* and *background knowledge attack*. Furthermore, anonymization techniques are traditional data publishing techniques proposed for a centralized database. As commented in a survey [46], these anonymization techniques in crowdsensing applications have a main drawback of the need for a trusted proxy to produce the anonymized values and send them to each participant. This implies the risk of single point of failure.

Cryptographic Methods. Cryptographic methods encrypt the training data before transmission. However, traditional cryptographic methods suffer from high computation complexity and the sophistication of key management [15]. The method of Homomorphic Encryption (HE) [12] does not need key propagation and has attracted research interest. With HE, computation on ciphertexts generates an encrypted result which matches the result of the operations performed on the plaintext data after decryption. In [20, 38, 47], the ML model is trained at the coordinator on the HE ciphertexts. During the inference phase, the data is also encrypted before transmission.

Limitations: However, in HE, operations on the ciphertexts are required to be expressed as polynomials of a bounded degree. Thus, HE is normally applied to the operations with a linear discrimination nature. Furthermore, HE involves intensive computation and leads to *data swelling*. As benchmarked in [26], HE causes computation overhead millions times higher than a multiplicative obfuscation approach that will be discussed shortly. Additionally, HE will make the training process at least an order of magnitude slower [16].

Data Obfuscation. Data obfuscation methods perturb the data samples used for training a global model. These methods include additive perturbation, multiplicative perturbation, and generative obfuscation.

- Additive Perturbation: Normally, additive obfuscation is often associated with Differential Privacy (DP) [13]. DP is a formal and quantifiable measure of privacy protection, which can be incorporated in data mining and data publishing [49]. The key idea of differentially private data mining [1, 9, 44] is to learn a model with plaintxt data but perturb the value computed in a certain step (e.g., gradients in optimization) with noises during the training. Such techniques, as discussed earlier, are often used in DML (§2.1.1). Differentially private data publishing aims to output aggregate information without revealing any specific entry. It can be achieved by adding noises to the query results using Laplacian [14], exponential [33], and median [39] mechanisms.
Limitations: Differentially private data publishing techniques are often used to support the release of limited data representations, such as contingency tables or histograms [49]. The amount of noise increases dramatically when the queries are correlated [49]. Besides, differentially private data publishing often caters into the setting of centralized systems, where a curator collects all the data and respond to queries. But such a trusted curator can be questionable and costly in the context of IoT. Moreover, it incurs the risk of single-point failure. If the curator is not available, each data contributor perturbs its own result, which leads to an aggregated noise that significantly exceeds the required amount to ensure ϵ-DP of the final result. The experiment in [26] shows that, when each participant independently perturbs the training data vectors with Laplacian noise to achieve ϵ-DP, the support vector macine and deep neural networks learned from the data yield poor accuracy.

- Multiplicative Perturbation: Random projection [10, 26, 29, 41] is a typical multiplicative perturbation. Some random projection schemes [10] preserve the dimensionality of the data but are susceptible to *approximate reconstruction attack* [30]. Other schemes [26, 29, 41] reduce the dimensions of the data to better preserve privacy. The approach in [41] standardizes the projection matrix R for all participants. However, this design may scale poorly since any collusion between participants would breach data privacy. In [26, 29], participants use different private matrices for random projection. Therefore, the Euclidean distances for the perturbed data are no longer preserved, which can significantly degrade the classification accuracy for distance-based classifiers. To tackle this problem, in [29], the coordinator uses regression to reconstruct the pairwise distances between the original data vectors based on each participant's obfuscated projection results of a set of public data samples. However, the coordinator can use the public samples and their projections to recover random projection matrix of each participant. In [26], rather than using regression for distance estimation, deep neural networks (DNNs) are leveraged to learn the sophisticated pattern of the projected data from multiple participants.
Limitations: In summary, there exists a trade-off between privacy and utility when applying multiplicative perturbation. If each participant uses a different random projection matrix, privacy can be better preserved but classification accuracy in the cloud is likely degraded.

- Generative Obfuscation: Different from additive/multiplicative perturbation techniques, generative models can also produce obfuscated data. Huang et al. [25] propose a Generative Adversarial Privacy (GAP) algorithm, which is composed of a *privatizer* and an *adversary* network. GAP formulates a minimax game-theoretic problem where the *privatizer* aims to obfuscate the original data X to render a specified privacy-sensitive attribute Y non-classifiable by the *adversary* network. The *privatizer* and *adversary* are trained in an iterative manner.

Limitations: Running a generative model locally for obfuscation incurs high computation overhead. As a data publishing technique, although GAP [25] restricts the l_2 distance between the original data vector and the obfuscated data vector to control the distortion level, there is no guarantee of the utility of the obfuscated data.

Data Synthesis. Data synthesis methods use generative models that capture the underlying distribution of a private dataset and generate resembling data samples. Such generalization, ideally, would protect individual-specific information. In [2], differentially private k-means clustering is applied on the raw dataset. Then, generative models are trained only on their own cluster using differentially private gradient descent [2].

Limitations: Data synthesis is a data publishing technique and is usually implemented in a centralized database with massive data samples. As generative models often incur high computation overhead, this approach is not suitable for resource-limited IoT devices.

2.2 Privacy-Preserving Inference

Compared with a body of research on privacy-preserving training, less work is dedicated to privacy-preserving inference. Privacy-preserving inference approaches assume that the ML model at the coordinator has been previously trained using public plaintext data. They aim to protect the privacy contained in test data vectors while maintaining the inference accuracy. Additive perturbation is generally not advisable for deep models because the inference accuracy of deep models can be significantly degraded by small perturbations on input data [48]. In order to achieve privacy preservation in the inference phase against an honest-but-curious coordinator running the ML model, CryptoNets [16] and Multi-party Computation (MPC) [4] are proposed.

2.2.1 CryptoNets. Gilad-Bachrach et al. [16] adjust the feed-forward neural network trained with plaintext data so that it can be applied to the homomorphically encrypted data to make encrypted inference. A secret key is needed to decrypt the result. Through the process, not only the data but also the inference result are kept secret against the honest-but-curious coordinator.

Limitations: Unfortunately, the high computational complexity of HE renders CryptoNets unpractical for IoT devices. Moreover, although CryptoNets does not need to support training over ciphertext, the neural network still needs to satisfy certain conditions. For example, a square polynomial function instead of a sigmoid or ReLU function should be used as the activation function. However, using square polynomial function as the activation function is rare for existing neural networks. Scaling is also required since encryption scheme does not support floating-point numbers.

2.2.2 Multi-Party Computation (MPC). MPC [17] enables the parties involved to jointly compute a function over their inputs while keeping those inputs private. Barni et al. [4] apply MPC in *privacy-preserving inference.* Specifically, the participant encrypts the data and sends it to the coordinator. The coordinator computes an inner product between the data and the weights of the first layer and sends the results back to the participant. Then, the participant applies decryption and non-linear transformation. Results are again encrypted before being transmitted to the coordinator. The process continues until all the layers have been computed. In this scheme, the input data and the knowledge embedded in the neural networks are both protected.

Limitations: MPC requires many rounds of communication between the participant and the coordinator, representing considerable communication overhead.

2.3 Remark

The existing approaches reviewed in §2.1 and §2.2 have different threat, privacy and system models. The anonymization and additive/multiplicative perturbation approaches often introduce affordable overheads and thus are feasible in the context of IoT. However, anonymization mainly focuses on private data publishing and does not address the problem of model training. Thus, additive and multiplicative perturbation approaches are promising for privacy-preserving training in IoT. In contrast, lightweight privacy-preserving inference approaches for IoT are lacking. Thus, we believe that privacy-preserving inference deserves more research attention. As such, §3 describes a lightweight and voluntary privacy-preserving inference approach called *ObfNet* and the preliminary results.

3 OBFNET: LIGHTWEIGHT & VOLUNTARY PRIVACY-PRESERVING INFERENCE

3.1 System, Threat, and Privacy Models

System model: The system consists of multiple resource-constrained participants and a resource-rich coordinator. The coordinator hosts a pre-trained deep model. The participants collect data samples and transmit them to the coordinator for inference using the deep model. The participants do not execute the deep model for inference due to the following reasons. First, each participant has limited computation and storage resources for executing the deep model. Second, in certain scenarios, the deep model may be commercially confidential and should not be released to the participants.

Threat model: The threat is an honest-but-curious coordinator. Specifically, the coordinator will not tamper with any data submitted by the participants and the inference results. However, the coordinator is curious about the private information contained in the data submitted by the participants.

Privacy model: The raw form of the submitted data is the participant's privacy to be protected. Data form confidentiality is an immediate and basic privacy requirement in many applications.

We discuss several related issues. First, although the inference result may also contain information about the participant, this paper does not consider label privacy. In practice, to mitigate the concern of label privacy leak, the participants can send the inference data samples to the coordinator via anonymity networks, such that the coordinator cannot associate the inference labels with the actual identities of the participants. Second, we aim to design a lightweight and voluntary approach to address the defined privacy threat. It is *lightweight* in that the computation performed by a participant should be affordable to resource-constrained IoT objects. It is *voluntary* in that any participant can choose to protect the data privacy by executing ObfNet for obfuscation or just send the original data, without needing to inform the coordinator. In other

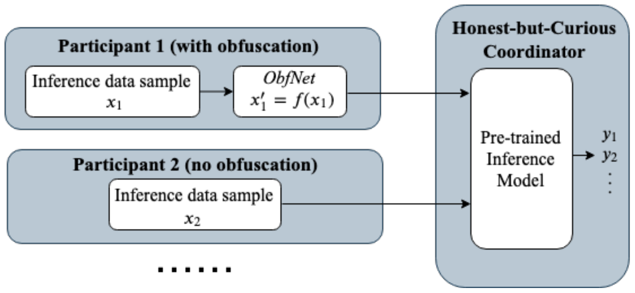
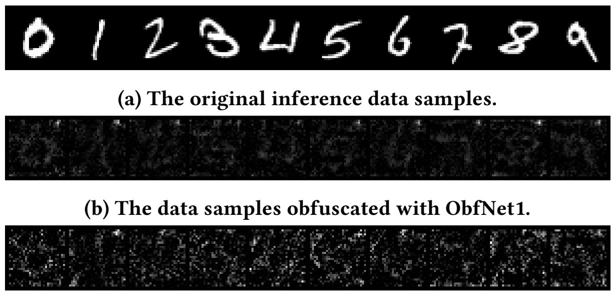

Participant 1 (with obfuscation)
Inference data sample x_1
ObfNet $x_1' = f(x_1)$
Participant 2 (no obfuscation)
Inference data sample x_2
Honest-but-Curious Coordinator
Pre-trained Inference Model
y_1
y_2

Figure 2: System model and approach overview.



(a) The original inference data samples.

(b) The data samples obfuscated with ObfNet1.

(c) The data samples obfuscated with ObfNet2-0.

Figure 3: Inference data samples from MNIST.

words, the pre-trained deep model at the coordinator admits both original and obfuscated data samples. This is a desirable feature for legacy deep models.

3.2 Construction of ObfNet

Fig. 2 illustrates our proposed ObfNet approach. The key idea is that we train a small-scale neural network called *ObfNet* that gives an output with the same size as the input. The training of ObfNet is as follows. After the pre-training of the deep inference model, we use the same training dataset to train the concatenation of the ObfNet and the pre-trained deep inference model. During the training phase of the concatenated model, only the parameters of the ObfNet are updated while the parameters of the pre-trained inference model are fixed. As discussed in §2, IoT end devices are in general incapable of ML model training. So, in our approach, the ObfNet is trained by the coordinator and released to the participants for use. The ObfNet uses many-to-one mapping activation functions, such as the rectified linear unit (ReLU). In that case, the coordinator cannot estimate the exact original data samples based on the obfuscated ones since there exist infinite possible inputs mapping to the same output. For better privacy protection, the coordinator may train multiple ObfNets and send all of them to each participant. The participants can randomly choose one to obfuscate the local data. Moreover, depending on the privacy-sensitivity of the data samples, participants can choose to obfuscate none or part of the data samples. The more ObfNets the coordinator trains and distributes, the less likely the coordinator can figure out which ObfNet is used. However, there exists a trade-off since training and distributing more ObfNets incur more communication overhead.

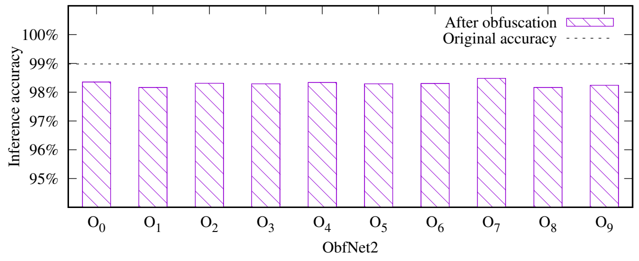


Figure 4: The inference accuracy of ObfNet2-x

3.3 Evaluation Results with MNIST Dataset

We evaluate our approach with the MNIST dataset [27]. The deep inference model is LeNet [27], with an inference accuracy of 98.98%. We train 11 different ObfNets denoted by ObfNet1 and ObfNet2-x, where x is from 0 to 9. ObfNet1 consists of one input layer of 784 neurons fully connected to an output layer of 784 neurons with ReLU as the activation function. It has 615,440 learnable parameters with a total volume of 2.17 MB. To show how the obfuscation results change with more hidden layers, we add one more hidden layer of 1,000 neurons in ObfNet2. ObfNet2-x, where x is from 0 to 9, are of the same structure and trained individually. Each has 1,569,784 learnable parameters and the total volume of each net is 6.01 MB. The LeNet and ObfNets are implemented using PyTorch [36].

Fig. 3 shows the original MNIST data samples and the obfuscated samples by ObfNet1 and ObfNet2-0. We can see that the obfuscated samples by ObfNet1 still contain some traces of the handwritten digits. However, these traces are not easily recognizable by human inspection. With one more hidden layer, the obfuscated samples by ObfNet2-0 are completely unrecognizable. Fig. 4 shows the inference accuracy of LeNet when each ObfNet2 is applied to obfuscate the data. As shown in the figure, the difference in the inference accuracy between each ObfNet2 is negligible and the average inference accuracy of the 10 ObfNets is 98.29%. Compared with the inference accuracy of LeNet on the original data, the accuracy drop is only 0.7%. Thus, the obfuscation maintains the inference accuracy well.

4 CONCLUSION AND FUTURE WORK

This paper reviews the existing privacy-preserving ML approaches that were developed largely in the context of cloud computing and discusses their limitations in the context of IoT. From our survey, there is a body of research on privacy-preserving training. Additive and multiplicative perturbation approaches are promising for privacy-preserving training in IoT due to their low computation and communication overhead. In contrast, lightweight privacy-preserving inference received limited research. Note that many IoT applications may prefer to use pre-trained deep models. Thus, how to protect the participants' data privacy in using the pre-trained deep models at the coordinator is a meaningful and interesting topic. To this end, we present ObfNet, a lightweight and voluntary privacy-preserving inference approach that obfuscates the data samples while maintaining the inference accuracy of an existing deep neural network. In our future work, we will perform more extensive evaluation of ObfNet including its performance with other datasets and overhead on actual IoT hardware platforms.

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 *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 308–318.

[2] Gergely Acs, Luca Melis, Claude Castelluccia, and Emiliano De Cristofaro. 2018. Differentially private mixture of generative neural networks. *IEEE Transactions on Knowledge and Data Engineering* 31, 6 (2018), 1109–1121.

[3] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. 2018. How to backdoor federated learning. *arXiv preprint arXiv:1807.00459* (2018).

[4] Mauro Barni, Claudio Orlandi, and Alessandro Piva. 2006. A privacy-preserving protocol for neural-network-based computation. In *Proceedings of the 8th workshop on Multimedia and security*. ACM, 146–151.

[5] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. 2018. Analyzing federated learning through an adversarial lens. *arXiv preprint arXiv:1811.12470* (2018).

[6] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. 2017. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1175–1191.

[7] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning* 3, 1 (2011), 1–122.

[8] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. 2017. Learning Differentially Private Recurrent Language Models. *arXiv preprint arXiv:1710.06963* (2017).

[9] Kamalika Chaudhuri and Claire Monteleoni. 2009. Privacy-preserving logistic regression. In *Advances in neural information processing systems*. 289–296.

[10] Keke Chen and Ling Liu. 2005. Privacy preserving data classification with rotation perturbation. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*. IEEE, 4–pp.

[11] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V Le, et al. 2012. Large scale distributed deep networks. In *Advances in neural information processing systems*. 1223–1231.

[12] Richard Alan DeMillo. 1978. *Foundations of secure computation.* Technical Report. Georgia Institute of Technology.

[13] Cynthia Dwork. 2011. Differential privacy. *Encyclopedia of Cryptography and Security* (2011), 338–340.

[14] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*. Springer, 265–284.

[15] Laurent Eschenauer and Virgil D Gligor. 2002. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM conference on Computer and communications security*. ACM, 41–47.

[16] Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. 2016. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *International Conference on Machine Learning*. 201–210.

[17] Oded Goldreich. 1998. Secure multi-party computation. *Manuscript. Preliminary version* 78 (1998).

[18] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[19] Google. [n.d.]. Google's Edge TPU. https://cloud.google.com/edge-tpu/.

[20] Thore Graepel, Kristin Lauter, and Michael Naehrig. 2012. ML confidential: Machine learning on encrypted data. In *International Conference on Information Security and Cryptology*. Springer, 1–21.

[21] Ralph Gross, Edoardo Airoldi, Bradley Malin, and Latanya Sweeney. 2005. Integrating utility into face de-identification. In *International Workshop on Privacy Enhancing Technologies*. Springer, 227–242.

[22] Ralph Gross, Latanya Sweeney, Fernando De la Torre, and Simon Baker. 2006. Model-based face de-identification. In *2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)*. IEEE, 161–161.

[23] Jihun Hamm, Adam C Champion, Guoxing Chen, Mikhail Belkin, and Dong Xuan. 2015. Crowd-ml: A privacy-preserving learning framework for a crowd of smart devices. In *2015 IEEE 35th International Conference on Distributed Computing Systems*. IEEE, 11–20.

[24] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. 2017. Deep models under the GAN: information leakage from collaborative deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 603–618.

[25] Chong Huang, Peter Kairouz, Xiao Chen, Lalitha Sankar, and Ram Rajagopal. 2017. Context-aware generative adversarial privacy. *Entropy* 19, 12 (2017), 656.

[26] Linshan Jiang, Rui Tan, Xin Lou, and Guosheng Lin. 2019. On Lightweight Privacy-Preserving Collaborative Learning for internet-of-things objects. (2019), 70–81.

[27] Yann LeCun. [n.d.]. The MNIST database of handwritten digits. *http://yann.lecun. com/exdb/mnist/* ([n. d.]).

[28] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. 2007. t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering*. IEEE, 106–115.

[29] Bin Liu, Yurong Jiang, Fei Sha, and Ramesh Govindan. 2012. Cloud-enabled privacy-preserving collaborative learning for mobile sensing. In *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*. ACM, 57–70.

[30] Kun Liu, Hillol Kargupta, and Jessica Ryan. 2005. Random projection-based multiplicative data perturbation for privacy preserving distributed data mining. *IEEE Transactions on knowledge and Data Engineering* 18, 1 (2005), 92–106.

[31] Ashwin Machanavajjhala, Johannes Gehrke, Daniel Kifer, and Muthuramakrishnan Venkitasubramaniam. 2006. l-diversity: Privacy beyond k-anonymity. In *22nd International Conference on Data Engineering (ICDE'06)*. IEEE, 24–24.

[32] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. 2016. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629* (2016).

[33] Frank McSherry and Kunal Talwar. 2007. Mechanism Design via Differential Privacy.. In *FOCS*, Vol. 7. 94–103.

[34] Elaine M Newton, Latanya Sweeney, and Bradley Malin. 2005. Preserving privacy by de-identifying face images. *IEEE transactions on Knowledge and Data Engineering* 17, 2 (2005), 232–243.

[35] Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2009), 1345–1359.

[36] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic Differentiation in PyTorch. In *NIPS Autodiff Workshop*.

[37] Le Trieu Phong, Yoshinori Aono, Takuya Hayashi, Lihua Wang, and Shiho Moriai. 2018. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security* 13, 5 (2018), 1333–1345.

[38] Yinian Qi and Mikhail J Atallah. 2008. Efficient privacy-preserving k-nearest neighbor search. In *2008 The 28th International Conference on Distributed Computing Systems*. IEEE, 311–319.

[39] Aaron Roth and Tim Roughgarden. 2010. Interactive privacy via the median mechanism. In *Proceedings of the forty-second ACM symposium on Theory of computing*. ACM, 765–774.

[40] Sandra Servia-Rodríguez, Liang Wang, Jianxin R Zhao, Richard Mortier, and Hamed Haddadi. 2018. Privacy-Preserving Personal Model Training. In *2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI)*. IEEE, 153–164.

[41] Yiran Shen, Chengwen Luo, Dan Yin, Hongkai Wen, Rus Daniela, and Wen Hu. 2018. Privacy-preserving sparse representation classification in cloud-enabled mobile applications. *Computer Networks* 133 (2018), 59–72.

[42] Reza Shokri and Vitaly Shmatikov. 2015. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. ACM, 1310–1321.

[43] Guocong Song and Wei Chai. 2018. Collaborative learning for deep neural networks. In *Advances in Neural Information Processing Systems*. 1832–1841.

[44] Shuang Song, Kamalika Chaudhuri, and Anand D Sarwate. 2013. Stochastic gradient descent with differentially private updates. In *2013 IEEE Global Conference on Signal and Information Processing*. IEEE, 245–248.

[45] Latanya Sweeney. 2002. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10, 05 (2002), 557–570.

[46] Idalides J Vergara-Laurens, Luis G Jaimes, and Miguel A Labrador. 2016. Privacy-preserving mechanisms for crowdsensing: Survey and research challenges. *IEEE Internet of Things Journal* 4, 4 (2016), 855–869.

[47] Justin Zhijun Zhan, LiWu Chang, and Stan Matwin. 2005. Privacy preserving k-nearest neighbor classification. *IJ Network Security* 1, 1 (2005), 46–51.

[48] Stephan Zheng, Yang Song, Thomas Leung, and Ian Goodfellow. 2016. Improving the robustness of deep neural networks via stability training. In *Proceedings of the ieee conference on computer vision and pattern recognition*. 4480–4488.

[49] Tianqing Zhu, Gang Li, Wanlei Zhou, and S Yu Philip. 2017. Differentially private data publishing and analysis: A survey. *IEEE Transactions on Knowledge and Data Engineering* 29, 8 (2017), 1619–1638.

[50] Martin Zinkevich, Markus Weimer, Lihong Li, and Alex J Smola. 2010. Parallelized stochastic gradient descent. In *Advances in neural information processing systems*. 2595–2603.