Efficient and Private Federated Learning using TEE

Fan Mo
Imperial College London
f.mo18@imperial.ac.uk

Hamed Haddadi
Imperial College London
h.haddadi@imperial.ac.uk

Extended Abstract

Background Federated Learning has received great attention since it enables edge devices to collaboratively train shared or personal models while keeping the raw training data local [3]. However, recent works have demonstrated that private information can leak from gradients present in shared models [4]. Due to the limited resources of edge devices, they do not support complicated defence methods against inference attacks. Therefore, guaranteeing data privacy during federated learning without compromising accuracy and efficiency is of great importance.

Recently, Trusted Execution Environments (TEE) have been used for privacy-preserving machine learning. By allocating private regions of computing resources (e.g. memory), it is possible to provide both hardware and software isolation. The TEE provides lower overhead and higher privacy compared to traditional software protections such as homomorphic encryption. After [5] first deployed deep neural networks (DNN) in a real TEE using Intel SGX technique, several studies added outsource support or multiple memory blacks to improve the efficiency of DNNs [2]. However, all these studies used advanced machines with Intel SGX as a setup to simulate high-performance computation, a solution that is hard to be extended in edge computing.

For federated learning on edge devices, such as mobile phones or small form factor platforms such as the Raspberry Pi, only limited computing resources are accessible. Leveraging a TEE implementation from ARM, TrustZone, we propose an efficient and private federated learning framework at edge computing. Differential privacy and data obliviousness will be used to enhance the privacy protection.

Framework

The proposed framework trains DNN models in the TrustZone to prevent the privacy leakage from model parameters. The layers are separated [6], and the DNN model $F()$ is partitioned as two parts. The untrusted part $F_U()$ is running in TrustZone's normal execution mode - the client application (CA). The trusted part $F_T()$ is running in TrustZone's trusted execution mode - the trusted application (TA).

We enhance the protection of the trusted part of models using data obliviousness. Side-channel attacks can track the sequence or patterns of operations such as the time, power, and memory address, and monitor trusted executions. To defend against tracking, data-oblivious algorithms relocate the memory address or obfuscate the path to access the memory address of the recently used value. Previous research [5] developed basic oblivious primitives such as comparisons and sorting by adding dummy memory addresses. Based on these oblivious operations, we enhance our framework to defend attacks that monitor memory usage patterns.

We use the differential privacy in stochastic gradient descent (SGD) algorithm [1] at the CA's layers and between the communication of the TA's layers and the CA's layers. The communication is vulnerable since transmitting parameters leaks information from the TA to the CA. Therefore, Gaussian noise source is added to obfuscate transmitted parameters. In order to minimize the cost, differential privacy is not applied to the complete trusted parts.

Initial experiment and results

To evaluate our framework, we used MNIST with Le-net and CIFAR-10 with a Small-net due to their minimum sizes that makes them suitable for federated learning on edge devices. In addition, we choose Open Portable TEE (OPTEE), an open source framework based on TrustZone, as the implementation. We use Darknet as the DNN framework because of its high performance and small dependencies, that makes it suitable for deploying in the TA. We conduct an initial experiment on a Raspberry Pi 3 Model B.

We succeeded on extracting any layer of models using our proposed framework. The initial results reveal that partitioning models leads to a slight decrease of the CPU usage in the user mode, though consequently, it increases the CPU usage in the kernel mode. Overall, the total cost of computation does not significantly increase along with the increasing number of layers in TrustZone.

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] Nick Hynes, Raymond Cheng, and Dawn Song. 2018. Efficient deep learning on multi-source private data. *arXiv preprint arXiv:1807.06689* (2018).

[3] Brendan McMahan and Daniel Ramage. 2017. Federated learning: Collaborative machine learning without centralized training data. *Google Research Blog* (2017).

[4] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. 2019. Exploiting Unintended Feature Leakage in Collaborative Learning. In *Proceedings of 40th IEEE Symposium on Security & Privacy*. IEEE, 480–495.

[5] Olga Ohrimenko, Felix Schuster, Cédric Fournet, Aastha Mehta, Sebastian Nowozin, Kapil Vaswani, and Manuel Costa. 2016. Oblivious Multi-Party Machine Learning on Trusted Processors.. In *USENIX Security Symposium*. 619–636.

[6] Seyed Ali Osia, Ali Taheri, Ali Shahin Shamsabadi, Minos Katevas, Hamed Haddadi, and Hamid RR Rabiee. 2019. Deep private-feature extraction. *IEEE Transactions on Knowledge and Data Engineering* (2019).