Robust and Actively Secure Serverless Collaborative Learning

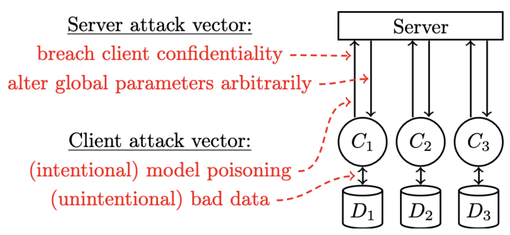
Olive Franzese, Adam Dziedzic, Christopher A. Choquette-Choo, Mark R. Thomas, Muhammad Ahmad Kaleem, Stephan Rabanser, Congyu Fang, Somesh Jha, Nicolas Papernot, Xiao Wang

University of Toronto, Vector Institute, Northwestern University, Google, University of Wisconsin-Madison, CISPA Helmholtz Center for Information Security

Introduction

- Collaborative machine learning (ML) is widely used to enable institutions to learn better models from distributed data.
- While collaborative approaches to learning intuitively protect user data, they remain vulnerable to either the server, the clients, or both, deviating from the protocol.
- We propose the first peer-to-peer (P2P) collaborative learning scheme that is secure against malicious servers and robust to malicious clients.
- Our novel generic framework transforms any (compatible) algorithm for robust aggregation of model updates to the setting where servers and clients can act maliciously.

Motivation



Contributions

- Current collaborative learning approaches are vulnerable to both client (denoted as C in the Figure above with data D) and server attack vectors. Our framework tackles all of these vulnerabilities simultaneously.
- We provide the first protocol that operates under the malicious threat model and is robust to malicious clients and servers. We provide a simulation-based proof of its cryptographic security.
- We design our protocol as a generic compiler that can convert broad categories of robust aggregation algorithms to our improved security model efficiently.

Problem Setup

- The P2P collaborative learning is conducted among a set of peers performing one of two roles: a client (or worker) who performs learning on a local dataset, or a server that aggregates the many client updates.
- A peer can be assigned either of the roles. The role of the server is performed by a subset of peers termed the aggregation committee.

Threat Model

- We consider a malicious threat model where clients and servers may perform arbitrary adversarial actions to interfere with the protocol.
- Malicious Clients may attempt to lower the quality of the trained model by sending distorted model updates. They may also attempt to steal information about the other peers' data, i.e. break confidentiality.
- Malicious Servers / Committee Members can arbitrarily modify model parameters and collude. For example, they may attempt to reconstruct individual data points from the clients' updates, thus breaking data confidentiality.

Robust and Actively Secure Framework

- Our framework efficiently lifts the robust aggregation algorithms (e.g., CC) to the P2P learning setting with guaranteed malicious-secure (or, actively-secure) protocol fidelity.
- We propose a modular design that encompasses a broad class of robust aggregation algorithms designed for the single-server setting.
- A key challenge that we surmount is strengthening security whilst maintaining the efficiency necessary to scale to real-world scenarios. To this end, we leverage computational subjectivity.

Framework Design

- We design a modular template to organize aggregation algorithms in terms of three functions F^C, F^P, and F^R.
- $F^C : \mathcal{D} \times \mathcal{S} \times \Omega \to U$ client-side update computation based on local data, state, and global model parameters; accordingly, $\mathcal{D}$ is the space of possible client datasets, $\mathcal{S}$ is the space of local states, Ω is the space of global model parameters, and U is the space of client updates.
- $F^P : U \to V$ server-side update preprocessing transforms each client update to a preprocessed domain V.
- $F^R : V^m \to \Omega$ server-side update aggregation, which combines the preprocessed local updates into a global model update $\boldsymbol{w} \in \Omega$.

Main Protocol Outline

1. The clients randomly select an aggregation committee $C \subset \{P_i\}_{i\in[m]}$.

Client update:

2. Each client P_i applies local computation $\boldsymbol{u}_i \leftarrow F^C(\mathsf{data}, \mathsf{st}, \boldsymbol{w})$.

Preprocessing:

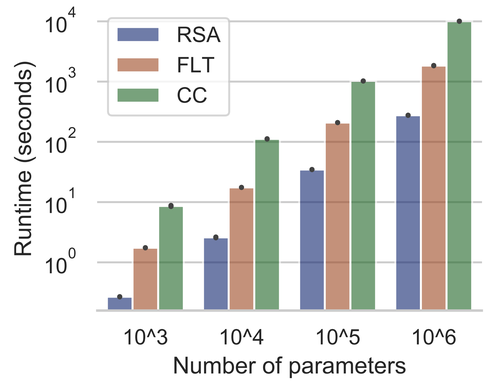
3. For each client P_i, compute $\boldsymbol{v}_i \leftarrow F^P(\boldsymbol{u}_i)$.
4. P_i secret shares $\boldsymbol{v}_i$ to obtain $[\boldsymbol{v}_i]$ and sends one share to each $P_j \in C$.
5. If F^P is not computationally surjective, P_i uses Distributed Zero Knowledge (DZK) to prove to the committee C that $\boldsymbol{v}_i$ is correctly computed from some $\boldsymbol{u}_i$ of P_i's choice. Otherwise, P_i uses DZK to prove that $\boldsymbol{v}_i \in V$.
6. If $\mathsf{Domain}(F^R) \neq \mathsf{Image}(F^P)$, P_i uses DZK to prove to the committee C that $\boldsymbol{v}_i \in \mathsf{Image}(F^P)$.

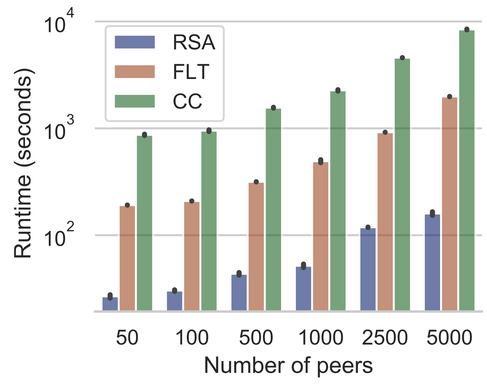
Update Aggregation:

7. All committee members $P_j \in C$ input shares $[\boldsymbol{v}_i]$ for all $i \in [n]$ to a $|C|$-party computation protocol in order to compute $\boldsymbol{w} \leftarrow F^R(\{\boldsymbol{v}_i\}_{i\in[n]})$. Committee members send $\boldsymbol{w}$ to all clients.

Empirical Evaluation

Computational Efficiency. The runtime performance of the robust aggregation algorithms (RSA, FLT, and CC) scales linearly with the number of parameters and peers





Security does not impact Robustness. We verify that the properties of the robust aggregation algorithms hold after the required modifications.