Fault-tolerant federated and distributed learning

Sanmi Koyejo
• ML models routinely trained/deployed in distributed settings
• Distributed learning useful for amortizing training costs, learning with physically distributed data.
• Distributed learning has implications for privacy
Centralized Learning

Distributed Learning

2: Local Computation

4: Aggregation

3: Push

1: Pull
Common strategies for distributed ML

1. **Distributed Training**
   - distributed gradient computation
   - server aggregates gradient updates

2. **Federated Learning**
   - distributed training on local data
   - server aggregates model parameters
Distributed ML is susceptible to failures:

- Hardware failures e.g. bit-flip computation errors
- Software failures e.g. label-flip errors
- Communication failures e.g. dropped updates
- Adversarial attacks (worst case): possibly targeted, coordinated training attacks
Robust Distributed SGD
Workers compute gradients on local data.
Threat Model

1: Pull

2: Gradient Computation

3: Push

Correct Update

Byzantine Update

Honest Worker

Honest Worker

Byzantine Worker

4: Aggregation

Server
Distributed SGD

\[ \min_x F(x) \quad \text{where} \quad F(x) = E_{z \sim D}[f(x; z)] \]

\( m \) workers, \( n \) samples per worker (wlog.)

\[ F_i(x) = \frac{1}{n} \sum_{j=1}^{n} f(x; z^{i,j}), \forall i \in [m] \]

Server update rule

\[ x^{t+1} = x^t - \gamma^t \text{Aggr}(\{g_i(x^t) : i \in [m]\}) \]

\[ g_i(x^t) = \begin{cases} \ast & \text{ith worker is faulty,} \\ \nabla F_i(x^t) & \text{otherwise,} \end{cases} \]
Compared to prior work

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Byzantine tolerance</th>
<th>Near-linear complexity $O(dm)$</th>
<th>Scalability</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$2q &lt; m$</td>
<td>$m \leq 2q &lt; 2m$</td>
<td></td>
</tr>
<tr>
<td>Krum$^1$</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trimmed mean$^2$ (median)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Zeno (our work)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

- $m$ workers
- $q$ malicious workers
- $d$ dimensional feature

Important to focus on learning convergence, not generic robustness

• Previous work on robust distributed learning (Median, Krum) has focused on Euclidean norm guarantees, roughly:

\[ \| g_t - E[\nabla F_t(x)] \| < \epsilon \]

• Note that norm robustness is less important than robustly estimating the descent direction

• Example: construct an attacker that satisfies norm guarantees, but is pointed in the wrong direction

Xie, K., Gupta “Fall of Empires: Breaking Byzantine-tolerant SGD by Inner Product Manipulation” (UAI 2019)
Breaking Robust Distributed Learning
Aggregation using Zeno

**Key idea:** Average the top-k gradients as sorted by *stochastic descendant score*

\[
Score_{\gamma, \rho}(u, x) = f_r(x) - f_r(x - \gamma u) - \rho \|u\|^2
\]

where \( f_r(x) = \frac{1}{n_r} \sum_{i=1}^{n_r} f(x; z_i) \)

- current model
- correct updates
- incorrect updates

Intuition: Correct updates establish a boundary (black dashed circle); Zeno lies inside the boundary
Zeno aggregation rule is robust

• Assumptions:
  • Stochastic descendant score estimate is unbiased
  • Loss function $f(x; z)$ is L-smooth and $\mu$-weakly convex
  • Variance of population gradient is bounded

• Sketch of main result (with up to q failed / malicious workers)

$$\sum_{t=0}^{T-1} \frac{E\|\nabla F(x_t)\|^2}{T} \leq \mathcal{O}\left(\frac{1}{\sqrt{T}}\right) + \mathcal{O}\left(\frac{(k-q+1)(m-q)}{(m-k)^2}\right)$$
5-layer CNN, CIFAR-10, bit-flipping attack, $m=20$

(a) Top-1 accuracy on testing set, with $q = 8$

(b) Cross entropy on training set, with $q = 8$

(c) Top-1 accuracy on testing set, with $q = 12$

(d) Cross entropy on training set, with $q = 12$
5-layer CNN, CIFAR-10, label-flipping attack, \(m=20\)

(a) Top-1 accuracy on testing set, with \(q = 8\)

(b) Cross entropy on training set, with \(q = 8\)

(c) Top-1 accuracy on testing set, with \(q = 12\)

(d) Cross entropy on training set, with \(q = 12\)
Robust Federated Learning
Is Federated Learning Simply Re-branded Distributed Learning?

- unbalanced, non-IID device data
- limited, heterogeneous device computation
- infrequent task scheduling
- limited, infrequent communication, congestion
- untrusted devices and data poisoning
Workers compute updated local model parameters.
Threat Model

1: Pull

2: Local Computation

Honest Worker

Byzantine Worker

3: Push

Correct Update

Byzantine Update

4: Aggregation
Compared to prior work

<table>
<thead>
<tr>
<th>Key property</th>
<th>Solution</th>
<th>By</th>
</tr>
</thead>
<tbody>
<tr>
<td>Limited computation</td>
<td>SGD</td>
<td>Previous work¹</td>
</tr>
<tr>
<td>Limited communication</td>
<td>Dropped updates</td>
<td></td>
</tr>
<tr>
<td>Private local data</td>
<td>Distributed (decentralized) training</td>
<td></td>
</tr>
<tr>
<td>Hardware, Software, Communication failures, Poisoned workers</td>
<td>Robust estimator</td>
<td>Our work</td>
</tr>
</tbody>
</table>

Federated Learning using Secure Local SGD

\[ \min_x F(x) \quad \text{where } F(x) = \frac{1}{n} \sum_{i=1}^{n} E_{z^i \sim D^i} [f(x; z^i)] \]

Device update: \( x_{t,h}^i \leftarrow x_{t,h-1}^i - \gamma \nabla f(x_{t,h-1}^i; z_{t,h}^i) \) [for \( H \) steps]

Server update: \( x'_t = \text{Trmean}_b \left( \{x_{t,H}^i : i \in S_t\} \right) \);
\[ x_t \leftarrow (1 - \alpha)x_{t-1} + \alpha x'_t \]

\( \text{Trmean}_b(\{u_i : i \in [l]\}) = \frac{1}{l-2b} \sum_{i=b+1}^{l-b} u_{\pi(i):\pi(l)} \)

\( \pi(\cdot) = \text{argsort}(\cdot) \quad S_t = \text{random subset of devices, } |S_t| = k \)
Proposed aggregation rule is robust

• Assumptions:
  • Existence of at least one global optimum (not necessarily unique)
  • Loss function \( f(x; z) \) is L-smooth and \( \mu \)-weakly convex
  • Variance of population gradient is bounded by \( V_1 \)

• Sketch of main result: With up to \( q \) failed/malicious devices, Federated learning convergence rate

\[
\sum_{t=0}^{T-1} \frac{E}{T} \| \nabla F(x^t) \|^2 \leq O \left( \frac{k(k+b)}{(k-b-q)^2} + \frac{1}{k-q} - \frac{1}{n} \right) + O \left( V_1 \right)
\]
5-layer CNN, CIFAR-10; Balanced data
100 workers; k=10; label-flipping attack; q=4 (per)

NOTE: SLSGD is equiv. to FedAvg when $\alpha = 1; b=0$. 

[Graph showing top-1 accuracy and loss over global epoch for different settings of $\gamma$, $\alpha$, and $b$.]
5-layer CNN, CIFAR-10; Unbalanced data
100 workers; k=10; label-flipping attack; q=4 (per)

NOTE: SLSGD is equiv. to FedAvg when \( \alpha = 1; b=0 \).
Careful aggregation is robust to worst-case failures

1. Suspicion-based aggregation for **distributed SGD**; robust to more than half adversarial workers

2. Regularized trimmed mean aggregation for **federated learning**; robust to non-IID data, communication failures, adversarial devices
Papers presented today


Some more light reading...


Asynchronous Federated ML

**Worker Side**
- Update local model using SGD on local loss regularized by global model

**Server Side**
- Scheduler thread to periodically trigger workers
- Update global model when updates received, with a discount factor proportional to staleness

Taken together, optimizes federated objective yet remains robust to delays, non-IID data, ...
5-layer CNN, CIFAR-10; 100 workers

Performance vs # Gradients Max staleness of 4, with Poly and Hinge temporal smoothing
5-layer CNN, CIFAR-10; Unbalanced data
100 workers; $k=10$; label-flipping attack; $q=4$ (per)

Performance vs # communication Max staleness of 4, with Poly and Hinge temporal smoothing
Thank you

sanmi@lillinois.edu
@sanmikoyejo