Learning with Strange Gradients

Martin Jaggi

EPFL

Machine Learning and Optimization Laboratory
mlo.epfl.ch
Collaborative & Federated Training

Updates

server or P2P

Data
1. Gradients from strange collaborators:
   - Federated Learning

2. Gradients from strange collaborators:
   - Personalization

3. Gradients from strange architectures

4. Gradients from faulty/malicious collaborators:
   - Byzantine-robust Training
Stochastic Gradient Descent (SGD)

\[
\min_x f(x) = \frac{1}{|\text{data}|} \sum_{i \in \text{data}} f_i(x)
\]

\[
i_t \sim \text{Uniform}(1, |\text{data}|)
\]

\[
x_{t+1} := x_t + \Delta x
\]

\[
\Delta x = -\gamma_t \nabla f_i(x_t)
\]

from backpropagation
Gradients from strange collaborators:
- Federated Learning
Client drift

- Federated Learning

\[
\min_x \frac{1}{n} \sum_{i}^{n} f_i(x)
\]

- Fed Avg / Local SGD

* for some local steps

\[
y_i := y_i - \eta \nabla f_i(y_i)
\]

\[
x := \frac{1}{n} \sum_{i=1}^{n} y_i \tag{aggregation}
\]
To reduce the bias in the local updates, we will apply this unbiased momentum every step. Updates are biased. To fix the former, we compute momentum using only the global optimizer state. Further, the server momentum is based on client data, optimizing only strategy (Server-only approach). However, such updates give rise to client drift. We next see how a different way of using momentum can mitigate client drift. In this section, we examine the tension between reducing communication by running multiple client updates each round, and degradation in performance due to client drift. A how momentum can help reduce client drift.
for some local steps

\[ y_i := y_i - \eta \left( (1 - \beta) \nabla f_i(y_i) + \beta m \right) \]

\[ m := (1 - \beta) \nabla f_i(x) + \beta m \]

aggregated on server after each round
Mime convergence

Number of rounds to reach
\[ \mathbb{E} \left[ \| \nabla f(x^{out}) \|^2 \right] \leq \varepsilon : \]

\[ \mathcal{O} \left( \left( \frac{n}{S} \right)^{3/2} \frac{L}{\varepsilon} \right) \] Scaffold

\[ \mathcal{O} \left( \frac{\delta(\zeta + \sigma)}{\varepsilon^{3/2}} + \frac{\zeta^2 + \sigma^2}{\varepsilon} + \frac{\delta}{\varepsilon} \right) \] MimeLiteMVR

\[ \mathcal{O} \left( \frac{\delta \zeta}{\sqrt{S} \varepsilon^{3/2}} + \frac{\zeta^2}{S \varepsilon} + \frac{\delta}{\varepsilon} \right) \] MimeMVR

\[ \Omega \left( \frac{L \zeta}{\sqrt{S} \varepsilon^{3/2}} + \frac{\zeta^2}{S \varepsilon} + \frac{L}{\varepsilon} \right) \] Lower bound (server-only)

Data Heterogeneity:
\( \delta \ll L \) inter-cl. Hessian similarity
\( \zeta \) inter-cl. gradient variance
\( \sigma \) intra-cl. gradient variance
Gradients from strange collaborators:
- Personalization
From federated towards decentralized
Collaborative Learning

- Federated

\[ \min_x \frac{1}{n} \sum_{i=1}^{n} f_i(x) \]

- Collaborative / Personalized

\[
\begin{align*}
\min_x f_0(x) & \quad \mid \\
\min_x f_1(x) & \quad \mid \\
\min_x f_n(x) & 
\end{align*}
\]
Personalized learning / optimization

- Weighted averaging

\[ x := x - \gamma \sum_{i=0}^{n} \alpha_i \nabla f_i(x) \]

- Weighted averaging with bias correction

\[ x := x - \gamma \sum_{i=0}^{n} (\alpha_i \nabla f_i(x) + c_i) \]

idea similar to Scaffold
Theorem: Convergence on personal objective $f_0$ for non-convex smooth objectives, using exponential moving average to learn $c_i$

$$\mathbb{E}[\|\nabla f_0(x^{out})\|^2] = O\left(\sqrt{\frac{LF_0 \sigma_0^2}{(n + 1)T}}\right)$$
Gradients from strange architectures
Alternating Partial Training for Neural Nets

Output Layer

Hidden Layers

Input Layer

$M_{\text{super}}$ $M_{\text{core}}$
**Theorem:** Convergence on original network $f$ for non-convex smooth objectives,

\[
\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E} \left[ \| \nabla f(x_t) \| ^2 \right] = \mathcal{O} \left( \sqrt{\frac{q^4 L F_0 \sigma^2}{T}} \right)
\]

and similarly for smaller core network

$q$: “gradient alignment” between parent and core network

- Masked Training of Neural Networks with Partial Gradients, arXiv
- AC/DC: Alternating Training of Deep Neural Networks, Peste et al, NeurIPS 2021
Gradients from faulty/malicious collaborators:
- Byzantine-robust Training
Malicious actors in FL

Updates

server

Unstable Client

Malicious Client
Byzantine Robust Training

\[ w := w - \gamma \text{agg}(\{g_i\}) \]

\[ \text{agg}(\{g_i\}) := \text{avg}(\{g_i\}) := \text{CM}(\{g_i\}) \]

Examples:
- Coordinate-wise median [Yin et al. 2017]
- Krum [Blanchard et al. 2018]
- Geometric median / RFA [Pillutla et al. 2019]
Robustness of the aggregation rule \( \text{agg}(\{g_i\}) \) does it imply robust training?

\[ \text{NO!} \]

Time-coupled attacks:

Little is enough
Strong negative result

- Any aggregation rule which does not use history will **fail** training (convergence)
Fix: Using history with momentum

- Simply use worker momentum
  \[ m_i := (1 - \beta)g_i + \beta m_i \]
- Effectively averages past gradients, reducing variance

- Aggregate worker momentum instead of gradients
  \[ w := w - \gamma \text{agg}(\{m_i\}) \]
Aggregation with Centered Clipping

- Norm-based clipping, before averaging

\[ CC = v + \text{clip}_\tau (g_i - v) \]

- Removes outliers

- Center at previous aggregated update
Robustness theorem

**Theorem:** Given any \((\delta_{\text{max}}, c)\)-robust aggregator, under a \(\delta\)-fraction of attackers and \(\sigma^2\) variance, our algorithm outputs \(x^{\text{out}}\) s.t.

\[
\mathbb{E} \| \nabla f(x^{\text{out}}) \|^2 \leq O \left( \sqrt{\frac{\sigma^2}{T}} \left( \delta + \frac{1}{n} \right) \right)
\]
References

1. Mime: Mimicking Centralized Stochastic Algorithms in Federated Learning
   - NeurIPS 2021  paper link

2. Optimal Model Averaging: Towards Personalized Collaborative Learning
   - FL workshop at ICML 2021  paper link

3. Linear Speedup in Personalized Collaborative Learning
   - arXiv  paper link

4. Masked Training of Neural Networks with Partial Gradients
   - arXiv  paper link

5. Learning from History for Byzantine Robust Optimization
   - ICML 2021  paper link
Thanks


EPFL
Machine Learning and Optimization Laboratory

website:  mlo.epfl.ch