Byzantine Resilient and Fast Federated Few-Shot Learning

Ankit Pratap Singh and Namrata Vaswani

Iowa State University

Multi-task representation learning/Few-shot Learning

First consider the centralized setting:

- Suppose that there are q source tasks.
- Each task $k \in [q]$ associated with a distribution over the input-output space $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \subseteq \Re^n$ and $\mathcal{Y} \subseteq \Re$.
- Each task observes $m < n$ samples from $\mathcal{X} \times \mathcal{Y}$.
- The aim is to learn prediction functions for all tasks simultaneously, leveraging a shared representation $\varphi : \mathcal{X} \to \mathcal{Z}$ that maps inputs to a Low-Dimensional feature space $\mathcal{Z} \subseteq \Re^r$ $(r < m)$.
- Few-shot learning refers to learning in data-scarce environment $(m < n)$.

Linear Model

Let the representation function class be Low-Dimensional Linear Representations i.e., $\{\mathbf{x} \mapsto \mathbf{U}^T\mathbf{x} | \mathbf{U} \in \Re^{n\times r}\}$ [1].

$$\begin{aligned}\mathbf{Y}_{m\times q} &= [(\mathbf{y}_1)_{m\times 1}, ..., (\mathbf{y}_q)_{m\times 1}] = [(\mathbf{X}_1)_{m\times n}(\theta_1^*)_{n\times 1}, ..., (\mathbf{X}_q)_{m\times n}(\theta_q^*)_{n\times 1}] \\ &= [(\mathbf{X}_1)_{m\times n}\mathbf{U}^*_{n\times r}(\mathbf{b}_1^*)_{r\times 1}, ..., (\mathbf{X}_q)_{m\times n}\mathbf{U}^*_{n\times r}(\mathbf{b}_q^*)_{r\times 1}]\end{aligned}$$

- The matrices $\mathbf{X}_k$s are independent and identically distributed (i.i.d.) over k.
- We assume that each $\mathbf{X}_k$ is a "random Gaussian" matrix, i.e., entry of it is i.i.d. standard Gaussian.
- The goal is to find the optimal representation φ^*, represented by $\mathbf{U}^*$.
- $\mathbf{b}_k^*$ is the new true linear predictor for all tasks $k \in [q]$.

[1] Du et al., Few-shot learning via learning the representation, provably

Solving this problem requires solving

$$\min_{\substack{\tilde{\mathbf{U}} \in \Re^{n \times r} \\ \tilde{\mathbf{B}} \in \Re^{r \times q}}} \sum_{k=1}^{q} \left\| \mathbf{y}_k - \mathbf{X}_k \tilde{\mathbf{U}} \tilde{\mathbf{b}}_k \right\|^2 \tag{1}$$

In interesting parallel works AltGDmin[2] and FedRep [3], a fast and communication-efficient GD-based algorithm was introduced for solving the mathematical problem given in (1).

[2] Nayer & Vaswani, Fast and sample-efficient federated low rank matrix recovery from column-wise linear and quadratic projections

[3] Collins et al., Exploiting Shared Representations for Personalized Federated Learning

- Use sample splitting: new indep set of samples for each update
- Factorize $\mathbf{\Theta} = \mathbf{UB}$, initialize $\mathbf{U}$ by spectral initialization (think of it as Federated PCA),
- alternate b/w minimization over $\mathbf{B}$ and (projected) GD for $\mathbf{U}$
- projected GD for $\mathbf{U}$

$$\mathbf{U}^+ \leftarrow \mathrm{QR}(\mathbf{U} - \eta \nabla_U f(\mathbf{U}, \mathbf{B}))$$

- AltGDmin and FedRep are two parallel works which are functionally equivalent.
- AltGDmin uses a better initialization than FedRep and hence also has a better sample complexity by a factor of r.

[4] Nayer & Vaswani, Fast and sample-efficient federated low rank matrix recovery from column-wise linear and quadratic projections

[5] Collins et al., Exploiting Shared Representations for Personalized Federated Learning

- In the federated setting, we assume that there are a total of L nodes. Each observes a different disjoint subset ($\widetilde{m} = m/L$) of rows of $\mathbf{Y}$. At most τL nodes can be Byzantine with $\tau < 0.4$. The nodes can only communicate with the center.

Byzantine attack is a "model update poisoning" attack where

1. It knows the full state of the center and every node (data and algorithm, including all algorithm parameters).
2. Different Byzantine adversaries can also collude.
3. They cannot modify the outputs of the other (non-Byzantine) nodes or of the center, or delay communication.

Byzantine nodes can thus design the worst possible attacks at each algorithm iteration.

Algorithm 1 Byz-Fed-AltGDmin-Learn: Complete algorithm

Nodes $\ell = 1, ..., L$

Compute $(\mathbf{U}_0)_\ell$ which is the matrix of top r left singular vectors of $(\hat{\mathbf{\Theta}}_0)_\ell := \sum_{k=1}^{q} (\mathbf{X}_k)_\ell{}^\top ((\mathbf{y}_k)_\ell)_{\mathrm{trunc}} \mathbf{e}_k^\top$

Key Idea 1: Subspace Median on $(\mathbf{U}_0)_\ell$'s

Central Server: Subspace Median

Orthonormalize: $\mathbf{U}_\ell \leftarrow QR((U_\ell)_0)$, $\ell \in [L]$

Compute $\mathcal{P}_{\mathbf{U}_\ell} \leftarrow \mathbf{U}_\ell \mathbf{U}_\ell^\top$, $\ell \in [L]$

Compute GM: $\mathcal{P}_{gm} \leftarrow \mathrm{GeometricMedian}\{\mathcal{P}_{\mathbf{U}_\ell}, \ell \in [L]\}$

Find $\ell_{best} = \arg\min_\ell \|\mathcal{P}_{\mathbf{U}_\ell} - \mathcal{P}_{gm}\|_F$

Output $\mathbf{U}_0 = \mathbf{U}_{out} = \mathbf{U}_{\ell_{best}}$

for $t = 1$ to T **do**

  Nodes $\ell = 1, ..., L$

  Set $\mathbf{U} \leftarrow \mathbf{U}_{t-1}$

  With $\mathbf{U}$ fixed, Least-Squares step over $(\mathbf{b}_k)_\ell$ for all k

  With $\mathbf{B}$ fixed, Gradient of $f(\mathbf{U}, \mathbf{B})$ w.r.t. $\mathbf{U}$: ∇f_ℓ

  Central Server

  Key Idea 2: Calculate GM of $\nabla f_\ell'$s

  $\nabla f^{GM} \leftarrow \mathrm{GeometricMedian}(\nabla f_\ell, \ell = 1, 2, \ldots L)$.

  Compute $\mathbf{U}^+ \leftarrow QR(\mathbf{U}_{t-1} - \frac{\eta}{\rho \tilde{m}} \nabla f^{GM})$

  return Set $\mathbf{U}_t \leftarrow \mathbf{U}^+$. Push $\mathbf{U}_t$ to nodes.

end for

Multi-task representation learning/Few-shot Learning

Theorem

(Byz-Fed-AltGDmin-Learn: Complete guarantee) Assume $\max_k \|\mathbf{b}_k^*\| \leq \mu\sqrt{r/q}\sigma_1(\boldsymbol{\Theta}^*)$ *for a constant* $\mu \geq 1$. *If*

$$\frac{m}{L} q \geq C\kappa^4\mu^2(n+q)r^2\log(1/\epsilon)$$

then, w.p. at least $1 - TLn^{-10}$,

$$\mathbf{SD}_F(\mathbf{U}^*, \mathbf{U}_T) \leq \epsilon$$

and $\|(\theta_k)_\ell - \theta_k^*\| \leq \epsilon\|\theta_k^*\|$ *for all* $k \in [q]$, $\ell \in [L]$. *The communication cost per node is order* $nr\log(\frac{n}{\epsilon})$. *The computational cost at any node is order* $nqr\log(\frac{n}{\epsilon})$ *while that at the center it is* $n^2L\log^3(Lr/\epsilon)$.

In solving this problem, we also introduce a novel secure solution to the federated subspace learning meta-problem that occurs in many different applications.
Estimate principal subspace $span(\mathbf{U}^*)$ of an unknown $n \times n$ symmetric matrix $\mathbf{\Phi}^*$ in a federated setting, while being resilient to **Byzantine Attacks**.

$$\mathbf{D}_{n\times q} = [(\mathbf{D}_1)_{n\times q_1}, ..., (\mathbf{D}_\ell)_{n\times q_\ell}, ..., (\mathbf{D}_L)_{n\times q_L}]$$

1. $\mathbf{U}^*$ is an $n \times r$ matrix denoting the top r eigenvectors of $\mathbf{\Phi}^*$
2. **Federated Setting:** Each node $\ell \in [L]$ observes a data matrix $\mathbf{D}_\ell$, that allows it
    - To estimate $\mathbf{\Phi}^*$ as $\mathbf{\Phi}_\ell = \mathbf{D}_\ell \mathbf{D}_\ell^\top / q_\ell$
    - To estimate $\mathbf{U}^*$ as $\mathbf{U}_\ell$, which are the top r eigenvectors of $\mathbf{\Phi}_\ell$

Algorithm: Subspace Median

Algorithm 2 Subspace Median

Input Subspace estimates $\hat{\mathbf{U}}_\ell$, $\ell \in [L]$.
Parameters T_{gm}

1: Orthonormalize: $\mathbf{U}_\ell \leftarrow QR(\hat{\mathbf{U}}_\ell)$, $\ell \in [L]$

2: Compute $\mathcal{P}_{\mathbf{U}_\ell} \leftarrow \mathbf{U}_\ell \mathbf{U}_\ell^\top$, $\ell \in [L]$

3: Compute GM: $\mathcal{P}_{gm} \leftarrow GM\{\mathcal{P}_{\mathbf{U}_\ell}, \ell \in [L]\}$

4: Find $\ell_{best} = \arg\min_\ell \|\mathcal{P}_{\mathbf{U}_\ell} - \mathcal{P}_{gm}\|_F$

5: Output $\mathbf{U}_{out} = \mathbf{U}_{\ell_{best}}$

Subspace-Median

Lemma

Suppose GM can be computed exactly and at least 60% $\mathbf{U}_\ell$'s satisfy

$$\mathbf{SD}_F(\mathbf{U}^*, \mathbf{U}_\ell) \leq \delta$$

then,

$$\mathbf{SD}_F(\mathbf{U}^*, \mathbf{U}_{out}) \leq 23\delta$$

- Including **probability argument**, If

  $$\Pr\left(\mathbf{SD}_F(\mathbf{U}^*, \mathbf{U}_\ell) \leq \delta\right) \geq 1 - p$$

  then,

  $$\Pr\left(\mathbf{SD}_F(\mathbf{U}^*, \mathbf{U}_{out}) \leq 23\delta\right) \geq 1 - \exp(-L\psi(0.4 - \tau, p))$$

  $$\psi(a, b) := (1 - a)\log\frac{1 - a}{1 - b} + a\log\frac{a}{b}$$

- If GM is approximated using using a linear time algorithm[6] then,

  $$\Pr\left(\mathbf{SD}_F(\mathbf{U}^*, \mathbf{U}_{out}) \leq 23\delta\right) \geq 1 - \mathbf{c_0} - \exp(-L\psi(0.4 - \tau, p))$$

[6] Cohen et al., Geometric median in nearly linear time

Resilient Federated PCA via Subspace Median of Means

In order to implement the "mean" step, we combine samples from $\rho = \frac{L}{\tilde{L}}$ ($\tilde{L} < L$) nodes by implementing $\tilde{L}$ different federated power methods.

Corollary

Assume that the set of Byzantine nodes remains fixed for all iterations and the size of this set is at most τL with $\tau < 0.4\tilde{L}/L$. If

$$\frac{q}{L} = \widetilde{q} \geq CK^4 \frac{{\sigma_1^*}^2}{\Delta^2} \frac{nr}{\epsilon^2} \cdot \frac{\tilde{L}}{L}$$

then, then, w.p. at least $1 - c_0 - \exp(-L\psi(0.4 - \tau, 2\exp(-n) + n^{-10}))$,

$$\mathbf{SD}_F(\mathbf{U}_{out}, \mathbf{U}^*) \leq \epsilon$$

Comparisons for solving the resilient federated PCA problem

Methods→	SVD-ResCovEst	ResPowMeth	SubsMed (Proposed)	PowMeth, no attack
Sample Comp for PCA (lower bound on q)	$\frac{n^2 L}{\epsilon^2}$	$\max\left(n^2 r^2, \frac{n}{\epsilon^2}\right) \cdot L$	$\frac{nrL}{\epsilon^2}$	$\frac{nr}{\epsilon^2}$
Communic Cost	n^2	$nr \frac{\sigma_r^*}{\Delta} \log(\frac{n}{\epsilon})$	nr	$nr \frac{\sigma_r^*}{\Delta} \log(\frac{n}{\epsilon})$
Compute Cost - node	$n^2 q_\ell$	$nq_\ell r \frac{\sigma_r^*}{\Delta} \log(\frac{n}{\epsilon})$	$nq_\ell r \frac{\sigma_r^*}{\Delta} \log(\frac{n}{\epsilon})$	$nq_\ell r \frac{\sigma_r^*}{\Delta} \log(\frac{n}{\epsilon})$
Compute Cost - center	$n^2 L \log^3\left(\frac{Ln}{\epsilon}\right)$	$nrL \frac{\sigma_r^*}{\Delta} \log(\frac{n}{\epsilon}) \log^3\left(\frac{Ln}{\epsilon}\right)$	$n^2 L \log^3\left(\frac{Ln}{\epsilon}\right)$	$nrL \frac{\sigma_r^*}{\Delta} \log(\frac{n}{\epsilon})$

- SVD-Resilient Covariance Estimation (SVD-ResCovEst): SVD on GM of Covariance matrices[7]
- Resilient Power Method (ResPowMeth): GM based modification of the power method[8]
- Baseline Power Method for a no-attack setting (PowMeth)

[7] Minsker, Geometric median and robust estimation in Banach spaces

[8] Hardt and Price, The noisy power method: A meta algorithm with applications

Thank You!