Communication-Efficient and Deferentially-Private Federated Learning via Sketching with Sharp Guarantees

Abstract

Fedearted learning...

1 Introduction

The main contributions of this paper are as follows:

-

2 Federated Learning with Sketching

Algorithm 1 `CS`: Count Sketch to compress $\mathbf{g} \in \mathbb{R}^d$.

Inputs: $\mathbf{g} \in \mathbb{R}^d, t, k, \mathbf{S}_{t\times k}, h_i(1 \leq i \leq t), sign_i(1 \leq i \leq t)$
Compress vector $\tilde{\mathbf{g}} \in \mathbb{R}^d$ into $\mathbf{S}(\tilde{\mathbf{g}})$:
for $\mathbf{g}_i \in \mathbf{g}$ **do**
  for $j = 1, \cdots, t$ **do**
    $\mathbf{S}[j][h_j(i)] = \mathbf{S}[j-1][h_{j-1}(i)] + \mathrm{sign}_j(i).\mathbf{g}_i$
  end for
end for
return $\mathbf{S}_{t\times k}$
Query $\mathbf{g}_S \in \mathbb{R}^d$ from $\mathbf{S}(\mathbf{g})$:
for $i = 1, \ldots, d$ **do**
    $\mathbf{S}_{\mathbf{g}} = \mathrm{Median}\{\mathrm{sign}_j(i).\mathbf{S}[j][h_j(i)] : 1 \leq j \leq t\}$
end for
Output: $\mathbf{S}(\mathbf{g})$

Algorithm 2 `HEAVYMIX` [1]

Inputs: $\mathbf{S}_{\mathbf{g}}$; parameter-k
Compress vector $\tilde{\mathbf{g}} \in \mathbb{R}^d$ into $\mathbf{S}(\tilde{\mathbf{g}})$:
Query $\hat{\ell}_2^2 = (1 \pm 0.5)\left\|\mathbf{g}\right\|^2$ from sketch $\mathbf{S}_{\mathbf{g}}$
$\forall j$ query $\hat{\mathbf{g}}_j^2 = \hat{\mathbf{g}}_j^2 \pm \frac{1}{2k}\left\|\mathbf{g}\right\|^2$ from sketch $\mathbf{S}_{\mathbf{g}}$
$H = \{j | \hat{\mathbf{g}}_j \geq \frac{\hat{\ell}_2^2}{k}\}$ and $NH = \{j | \hat{\mathbf{g}}_j < \frac{\hat{\ell}_2^2}{k}\}$
$\mathrm{Top}_k = H \cup rand_\ell(NH)$, where $\ell = k - |H|$
Second round of communication to get exact values of Top_k
Output: $\mathbf{g}_S : \forall j \in \mathrm{Top}_k : \mathbf{g}_{Si} = \mathbf{g}_i$ and $\forall \notin \mathrm{Top}_k : \mathbf{g}_{Si} = 0$

Algorithm 3 `PFL`(R, τ, η, γ): Private Federated Learning with Sketching for homogeneous setting.

Inputs: $\boldsymbol{w}^{(0)}$ as an initial model shared by all local devices, the number of communication rounds R, the the number of local updates τ, and global and local learning rates γ and η, respectively
for $r = 0, \ldots, R-1$ **do**
  parallel for device $j = 1, \ldots, n$ **do**:
    Set $\boldsymbol{w}_j^{(0,r)} = \boldsymbol{w}^{(r-1)} - \gamma \mathbf{S}^{(r-1)}$
    for $c = 0, \ldots, \tau - 1$ **do**
      Sample a mini-batch $\xi_j^{(\ell,r)}$ and compute $\tilde{\mathbf{g}}_j^{(\ell,r)} \triangleq \nabla f_j(\boldsymbol{w}_j^{(\ell,r)}, \xi_j^{(c,r)})$
      $\boldsymbol{w}_j^{(\ell+1,r)} = \boldsymbol{w}_j^{(\ell,r)} - \eta\ \tilde{\mathbf{g}}_j^{(c,r)}$
    end for
      Device j sends $\mathbf{S}\left(\boldsymbol{w}_j^{(0,r)} - \ \boldsymbol{w}_j^{(\tau,r)}\right)$ back to the server.
  Server **computes**
      $\mathbf{S}^{(r)} = \frac{1}{p}\sum_{j=1} \mathbf{S}\left(\boldsymbol{w}_j^{(0,r)} - \ \boldsymbol{w}_j^{(\tau,r)}\right)$ and **broadcasts** $\mathbf{S}^{(r)}$ to all devices.
  end parallel for
end
Output: $\boldsymbol{w}^{(R-1)}$

Algorithm 4 `PFLGT`(R, τ, η, γ): Private Federated Learning with Sketching and gradient tracking.

Inputs: $\boldsymbol{w}^{(0)}$ as an initial model shared by all local devices, the number of communication rounds R, the the number of local updates τ, and global and local learning rates γ and η, respectively
for $r = 0, \ldots, R-1$ **do**
  parallel for device $j = 1, \ldots, n$ **do**:
    Set $\mathbf{c}_j^{(r)} = \mathbf{c}_j^{(r-1)} - \frac{1}{\tau}\left(\mathbf{S}^{(r-1)} - \mathbf{S}_j^{(r-1)}\right)$
    Set $\boldsymbol{w}_j^{(0,r)} = \boldsymbol{w}^{(r-1)} - \gamma \mathbf{S}^{(r-1)}$
    for $\ell = 0, \ldots, \tau - 1$ **do**
      Sample a minibatch $\xi_j^{(\ell,r)}$ and compute $\tilde{\mathbf{g}}_j^{(\ell,r)} \triangleq \nabla f_j(\boldsymbol{w}_j^{(\ell,r)}, \xi_j^{(\ell,r)})$
      $\boldsymbol{w}_j^{(\ell+1,r)} = \boldsymbol{w}_j^{(\ell,r)} - \eta\ \left(\tilde{\mathbf{g}}_j^{(\ell,r)} - \mathbf{c}_j^{(r)}\right)$
    end for
      Device j sends $\mathbf{S}_j^{(r)} \triangleq \mathbf{S}\left(\boldsymbol{w}_j^{(0,r)} - \ \boldsymbol{w}_j^{(\tau,r)}\right)$ back to the server.
  Server **computes**
      $\mathbf{S}^{(r)} = \frac{1}{p}\sum_{j=1} \mathbf{S}_j^{(r)}$ and **broadcasts** $\mathbf{S}^{(r)}$ to all devices.
  end parallel for
end
Output: $\boldsymbol{w}^{(R-1)}$

Algorithm 5 `CFL`$(R,\, \tau, \eta, \gamma)$: Communication-efficient Federated Learning with Sketching for homogeneous setting.

Inputs: $\boldsymbol{w}^{(0)}$ as an initial model shared by all local devices, the number of communication rounds R, the the number of local updates τ, and global and local learning rates γ and η, respectively
for $r = 0, \dots, R-1$ **do**
  parallel for device $j = 1, \dots, n$ **do**:
    Set $\boldsymbol{w}_j^{(0,r)} = \boldsymbol{w}^{(r-1)} - \gamma \underline{\mathbf{S}}^{(r)}$
    for $c = 0, \dots, \tau - 1$ **do**
      Sample a mini-batch $\xi_j^{(\ell,r)}$ and compute $\tilde{\mathbf{g}}_j^{(\ell,r)} \triangleq \nabla f_j(\boldsymbol{w}_j^{(\ell,r)}, \xi_j^{(c,r)})$
      $\boldsymbol{w}_j^{(\ell+1,r)} = \boldsymbol{w}_j^{(\ell,r)} - \eta\, \tilde{\mathbf{g}}_j^{(c,r)}$
    end for
      Device j sends $\mathbf{S}_j^{(r)} = \mathbf{S}\left(\boldsymbol{w}_j^{(0,r)} - \boldsymbol{w}_j^{(\tau,r)}\right)$ back to the server.
  Server **computes**
      $\mathbf{S}^{(r)} = \frac{1}{p} \sum_{j=1}^{n} \mathbf{S}\left(\boldsymbol{w}_j^{(0,r)} - \boldsymbol{w}_j^{(\tau,r)}\right)$
      Sever runs $\underline{\mathbf{S}}^{(r)} = \texttt{HEAVYMIX}(\mathbf{S}^{(r)})$.
  end parallel for
end
Output: $\boldsymbol{w}^{(R-1)}$

Algorithm 6 `CFL`$(R,\, \tau, \eta, \gamma)$: Communication-efficient Federated Learning with Sketching and gradient tracking.

Inputs: $\boldsymbol{w}^{(0)}$ as an initial model shared by all local devices, the number of communication rounds R, the the number of local updates τ, and global and local learning rates γ and η, respectively
for $r = 0, \dots, R-1$ **do**
  parallel for device $j = 1, \dots, n$ **do**:
    Set $\boldsymbol{w}_j^{(0,r)} = \boldsymbol{w}^{(r-1)} - \gamma \underline{\mathbf{S}}^{(r)}$
    Set $\mathbf{c}_j^{(r)} = \mathbf{c}_j^{(r-1)} - \frac{1}{\tau}\left(\underline{\mathbf{S}}^{(r)} - \underline{\mathbf{S}}_j^{(r)}\right)$
    for $c = 0, \dots, \tau - 1$ **do**
      Sample a mini-batch $\xi_j^{(\ell,r)}$ and compute $\tilde{\mathbf{g}}_j^{(\ell,r)} \triangleq \nabla f_j(\boldsymbol{w}_j^{(\ell,r)}, \xi_j^{(c,r)})$
      $\boldsymbol{w}_j^{(\ell+1,r)} = \boldsymbol{w}_j^{(\ell,r)} - \eta\, \tilde{\mathbf{g}}_j^{(c,r)}$
    end for
      Device j sends $\mathbf{S}_j^{(r)} = \mathbf{S}\left(\boldsymbol{w}_j^{(0,r)} - \boldsymbol{w}_j^{(\tau,r)}\right)$ back to the server.
  Server **runs**
      $\underline{\mathbf{S}}_j^{(r)} = \texttt{HEAVYMIX}(\mathbf{S}_j^{(r)})$ and returns $\underline{\mathbf{S}}_j^{(r)}$ to server j.
      $\underline{\mathbf{S}}^{(r)} = \frac{1}{p} \sum_{j=1}^{n} \underline{\mathbf{S}}_j^{(r)}$
      Sever broadcasts $\underline{\mathbf{S}}^{(r)}$.
  end parallel for
end
Output: $\boldsymbol{w}^{(R-1)}$

3 Convergence Analysis

3.1 Assumptions

4 Experiments

5 Conclusion

References

[1] N. Ivkin, D. Rothchild, E. Ullah, I. Stoica, R. Arora *et al.*, "Communication-efficient distributed sgd with sketching," in *Advances in Neural Information Processing Systems*, 2019, pp. 13 144–13 154.