1 紹介論文

Byzantine-Robust Distributed Learning Towards Optimal Statistical Rates

Yin, Dong and Chen, Yudong and Kannan, Ramchandran and Bartlett, Peter

In *Proceedings of the 35th International Conference on Machine Learning*, pages 5650–5659, 2018 (ICML2018)

1.1 Introduction

近年, 利用可能なデータの増加に伴い, 複雑な予測モデルが必要とされている. それにより, 処理の分散化が注目されている. 大規模の分散システムではロバストネスやセキュリティが非常に重要になってくる. 実際に, 個々のコンピュータが異常な動作をする可能性がある (クラッシュ, ハードウェアの障害, 計算停止,etc.). この論文ではフェデレーション ラーニング (学習をクラウド上ではなく, ユーザのモバイル端末上で実施するというもの) についてのロバストネスを議論する. さらに異常な動作としてビザンチン障害を考える. この論文ではビザンチン障害に耐えられる分散学習アルゴリズムを開発することを目指す.

目的は損失関数を最小化するパラメータでモデルを学習することである.m は分散された各ワーカーの数,αm はビザンチンワーカーの数,n は各ワーカーが持っているデータ数とする. 少なくとも強凸な損失関数では以下よりもエラーレートを小さくすることは不可能である.($\widetilde{\Omega}$ は α, n, m に独立な要素,n, m の対数を排除したものである.)

$$\widetilde{\Omega}\left(\frac{\alpha}{\sqrt{n}}+\frac{1}{\sqrt{nm}}\right)$$

また, 分散システムでは, 通信効率も考慮しなければならない. 各マシンの通信にはコストがかかる. だからラウンドごとに通信するデータは少量にするべきである. 今回は各ラウンドでベクトルのサイズ $\mathcal{O}(d)$ (d は学習されるパラメータの次元) しか通信できないとする.

つまり以下の二つを達成するアルゴリズムを考える.

- 統計的最適性:$\widetilde{\mathcal{O}}\left(\frac{\alpha}{\sqrt{n}}+\frac{1}{\sqrt{nm}}\right)$ を得る
- 通信効率:$\mathcal{O}(d)$ を満たす

1.1.1 Contributions

2 つの GD アルゴリズムを提案する.1 つは座標中央値に基づくアルゴリズムであり, もう一つは座標方向にトリム平均するアルゴリズムである. さらに 1 回の通信ラウンドのみを必要とするアルゴリズムを提案する. 以上 3 つのアルゴリズムを提案し, 以下のエラーレートが得られた.

- Median-based GD:$\widetilde{\mathcal{O}}\left(\frac{\alpha}{\sqrt{n}}+\frac{1}{\sqrt{nm}}+\frac{1}{n}\right)$
- Trimmed-mean-based GD:$\widetilde{\mathcal{O}}\left(\frac{\alpha}{\sqrt{n}}+\frac{1}{\sqrt{nm}}\right)$
- Median-based one-round algrithm:$\widetilde{\mathcal{O}}\left(\frac{\alpha}{\sqrt{n}}+\frac{1}{\sqrt{nm}}+\frac{1}{n}\right)$

1.2 Related Work

分散型ビザンチン障害という問題設定で median-of-means(データ m の部分集合に分割し, 各部分集合から推定値を計算し, その中央値を取る) という手法は [1] と [2] で用いられている. しかし [1] では単に 1-shot で用いられている, またエラーレートが最適ではない.[2] では損失関数が強凸の場合しか考慮されていない. またエラーレートが最適ではない.

[3] はこの論文と最も近い問題設定である (krum の論文). しかし [3] は, 無制限で独立したアクセスをする SGD で勾配を計算している点で異なる (?). また, エラーレートは示されていない.

1.3 Problem Setup

訓練データ $\mathbf{z}$ は未知の分布 $\mathcal{D}$ に従う. パラメータベクトル $\mathbf{w} \in \mathcal{W} \subseteq \mathbb{R}^d$ とすると, 損失関数は $F(\mathbf{w}) = \mathbb{E}_{\mathbf{z}}\{f(\mathbf{w};\mathbf{z})\}$ となる. 目標は以下のパラメータで学習することである.

$$\mathbf{w}^* = \arg\min_{\mathbf{w}\in W} F(\mathbf{w})$$

パラメータ空間 $\mathcal{W}$ は凸であり,

$$\|\mathbf{w} - \mathbf{w}'\|_2 \leq D,\ \forall \mathbf{w}, \mathbf{w}' \in \mathcal{W}$$

を満たすとする.

$z^{i,j}$ を i 番目のワーカーの j 番目のデータとすると $F_i(\mathbf{w}) = \frac{1}{n}\sum_{j=1}^{n} f(\mathbf{w};\mathbf{z}^{i,j})$ は i 番目のワーカーの経験損失関数になる.m 人のワーカーの中に αm 人のビザンチンワーカーがいて,$(1-\alpha)m$ 人の正しいワーカーがいる. ビザンチンワーカーは完全な知識を持っており, お互いが協力できるため任意の値をサーバーに送ることができる.

1.4 Model

Definition 1 では各ワーカーが計算した中央値をサーバに送る.Definition 2 では各ワーカーが計算したトリム平均をサーバに送る. そしてサーバがパラメータ更新して, 更新したパラメータを各ワーカーに渡す. 以上の操作を繰り返す.

Definition 1 (Coordinate-wise median(Option Ⅰ))**.** $\mathbf{x}^i \in \mathbb{R}^d, i \in [m]$ とする.the coordinate-wise median($\mathbf{g} = \text{med}\ \{x^i : i \in [m]\}$) は $k \in [d]$ 番目の要素に $g_k = \text{med}\ \{x_k^i : i \in [m]\}$ を持つベクトルである.

Definition 2 (Coordinate-wise trimmed mean(Option Ⅱ))**.** $\beta \in [0, \frac{1}{2}), \mathbf{x}^i \in \mathbb{R}^d, i \in [m]$, とする.the coordinate-wise β-trimmed mean($\mathbf{g} = \text{trmean}_\beta\ \{x^i : i \in [m]\}$) は $k \in [d]$ 番目の要素に $g_k = \frac{1}{(1-2\beta)m}\sum_{x\in U_k} x$ U_k は $\{x_k^1, \ldots, x_k^m\}$ から 1 β 番目まで大きい要素と小さい要素を取り除いた部分集合とする.

次に通信コストを抑えつつ, ロバストな推定をすることを考える.one-round アルゴリズムを提案する. 各ワーカーそれぞれがパラメータ更新を行い,$\widehat{\mathbf{w}}^i = \arg\min_{\mathbf{w}\in W} F_i(\mathbf{w})$ を求めて, パラメータをサーバに送り, サーバは中央値を計算するアルゴリズムである. ワーカーとサーバの通信回数は 1 回であるので通信コストはかなり抑えることができる.

Algorithm 1 Robust Distributed Gradient Descent

Require: Initialize parameter vector $\mathbf{w}^0 \in \mathcal{W}$, algorithm parameters β (for Option II), η and T.
for $t = 0, 1, 2, \ldots, T-1$ **do**
Master machine: send $\mathbf{w}^t$ to all the worker machines.
for $i \in [m]$ **in parallel do**
Worker machine i: compute local gradient

$$\mathbf{g}^i(\mathbf{w}^t) \leftarrow \begin{cases} \nabla F_i(\mathbf{w}^t) & \text{normal machines,} \\ * & \text{Byzantine machines,} \end{cases}$$

send $\mathbf{g}^i(\mathbf{w}^t)$ to master machine.
end for
Master machine: compute aggregate gradient

$$\mathbf{g}(\mathbf{w}^t) \leftarrow \begin{cases} \mathsf{med}\{\mathbf{g}^i(\mathbf{w}^t) : i \in [m]\} & \text{Option I} \\ \mathsf{trmean}_\beta\{\mathbf{g}^i(\mathbf{w}^t) : i \in [m]\} & \text{Option II} \end{cases}$$

update model parameter $\mathbf{w}^{t+1} \leftarrow \Pi_{\mathcal{W}}(\mathbf{w}^t - \eta \mathbf{g}(\mathbf{w}^t))$.
end for

また, エラーレート $\widetilde{\mathcal{O}}\left(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n}\right)$ が得られる. しかし損失関数が quadratic のみしかエラーレートの保障がされていない. したがって, 少なくても損失関数が quadratic の場合,one-round アルゴリズムは通信コストが大幅に少ないので有用である.

Definition 9 (Quadratic loss function)**.** 損失関数 $f(\mathbf{w}; \mathbf{z})$ は以下を満たすならば quadratic である.

$$f(\mathbf{w}; \mathbf{z}) = \frac{1}{2}\mathbf{w}^T\mathbf{H}\mathbf{w} + \mathbf{p}^T\mathbf{w} + c$$

$$\mathbf{z} = (\mathbf{H}, \mathbf{p}, c)$$

Algorithm 2 Robust One-round Algorithm

for $i \in [m]$ **in parallel do**
Worker machine i: compute & send to master machine:

$$\widehat{\mathbf{w}}^i \leftarrow \begin{cases} \arg\min_{\mathbf{w} \in \mathcal{W}} F_i(\mathbf{w}) & \text{normal machines} \\ * & \text{Byzantine machines} \end{cases}$$

end for
Master machine: compute $\widehat{\mathbf{w}} \leftarrow \mathsf{med}\{\widehat{\mathbf{w}}^i : i \in [m]\}$.

1.5 Convergence

論文中では, 各アルゴリズムのエラーレートの算出は損失関数が強凸, 凸, 非凸の場合に分けて証明されている. 今回は Median-based Gradient Descent の強凸の場合を扱う.

Assumption 1 (Smoothness of f and F)**.** $\partial_k f(\cdot, z)$ が L_k-Lipschitz である $(k \in [d])$. また,$f(\cdot, z)$ が L-smooth であると仮定する. さらに $F(\cdot)$ は L_F-smooth だと仮定する. また $\widehat{L} = \sqrt{\sum_{k=1}^d L_k^2}$ とする.

Assumption 2 (minimizer in W)**.** $F(\cdot)$ が凸である,$\mathbf{w}^* = \arg\min_{\mathbf{w}\in W} F(\mathbf{w})$ であるとし,$\nabla F(\mathbf{w}^*) = 0$ とする.

Assumption 3 (Bounded variance of gradient)**.**

$$\mathrm{Var}(\nabla f(\mathbf{w};\mathbf{z})) \leq V^2 \ , \forall \mathbf{w}$$

Assumption 4 (Bounded skewness of gradient)**.**

$$\|\gamma(\nabla f(\mathbf{w};\mathbf{z}))\|_\infty \leq S \ , \forall \mathbf{w}$$

Assumption 5 (Size of $\mathcal{W}$)**.** $\mathcal{W}$ は w^* を中心とする l_2-ball$\{w \in \mathbb{R}^d : \|\mathbf{w} - \mathbf{w}^*\|_2 \leq 2\|\mathbf{w}^0 - \mathbf{w}^*\|_2\}$ を含んでいる.

Theorem 1 (Strongly convex in Median-based Gradient Descent)**.** $F(\cdot)$ が λ_F-strongly convex であり,Assumption1〜4 を満たし,$\eta = \frac{1}{L_F}$ として,α が (1) を満たすならば, ならば以下が成り立つ.

$$\alpha + \sqrt{\frac{d\log(1 + nm\widehat{L}D)}{m(1-\alpha)}} + 0.4748\frac{S}{\sqrt{n}} \leq \frac{1}{2} - \varepsilon \tag{1}$$

少なくとも確率 $1 - \frac{4d}{1+nm\widehat{L}D^d}$ で T ラウンド後に

$$\|\mathbf{w}^T - \mathbf{w}^*\|_2 \leq \left(1 - \frac{\lambda_F}{L_F + \lambda_F}\right)^T \|\mathbf{w}^0 - \mathbf{w}^*\|_2 + \frac{2}{\lambda_F}\triangle$$

が成り立つ.

$$\triangle = \mathcal{O}\left(C_\epsilon V(\frac{\alpha}{\sqrt{n}} + \sqrt{\frac{d\log(nm\widehat{L}D)}{nm}} + \frac{S}{n})\right)$$

$$C_\epsilon = \sqrt{2\pi}\exp\left(\frac{1}{2}(\Phi^{-1}(1-\epsilon))^2\right)$$

$T \leq \frac{L_F+\lambda_F}{\lambda_F}\log(\frac{\lambda_F}{2\triangle}\|\mathbf{w}^0 - \mathbf{w}^*\|_2)$ のとき, 高い確率で誤差 $\|\widehat{\mathbf{w}} - \mathbf{w}^*\|_2 \leq \frac{4}{\lambda_F}\triangle$ を満たす $\widehat{\mathbf{w}} = \mathbf{w}^T$ を得ることができる.

1.6 Comparisons(trimmed mean and median)

trimmed mean GD と median は互いに補完的である. エラーレートは trimmed mean の方が優れているが, 損失関数の微分値の仮定は trimmed mean の方が厳しい. また,trimmed mean は追加のパラメータ β を必要とする.

	median GD	trimmed mean GD
Rate	$\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$	$\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$
$\partial_k f(\mathbf{w};\mathbf{z})$	Bounded skewness	Sub-exponential
α known?	No	Yes

Table 1. Comparison between the two robust distributed gradient descent algorithms.

1.7 Experiments

MNIST データセットで実験する. ビザンチンワーカーはラベル y の訓練データを $9-y$ に置き換えて学習する. ビザンチンワーカーは極端に変な値を送るのではなく, 適度に修正した値を送る.

多クラスロジスティック回帰モデルと CNN を使って学習する.CNN では SGD で勾配を計算する (各ワーカーの訓練データの 10% を用いる).

one-round algorithm では quadratic のみにしか理論的保障はないが, 実際にはロジスティック損失のような quadratic でない他の損失関数でもテスト誤差の改善ができる.

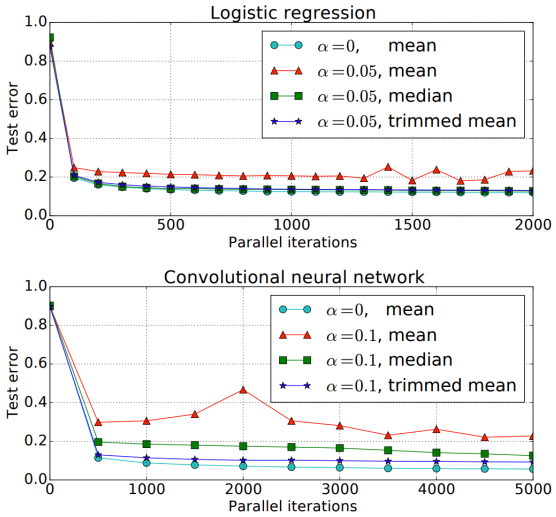

Figure1 Test error vs the number of parallel iterations. For logistic regression, we set m = 40, and for trimmed mean, we choose $\beta = 0.05$; for CNN, we set m = 10, and for trimmed mean, we choose $\beta = 0.1$

α	0	0.1	
Algorithm	mean	mean	median
Acc. (%)	91.8	83.7	89.0

Table 4. Test accuracy on the logistic regression model using one-round algorithm. We set $m = 10$.

References

[1] Feng, J., Xu, H., and Mannor, S. Distributed robust learning. arXiv preprint arXiv:1409.5937, 2014.

[2] Chen, Y., Su, L., and Xu, J. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. arXiv preprint arXiv:1705.05491, 2017.

[3] Blanchard, P., Mhamdi, E. M. E., Guerraoui, R., and Stainer, J. Machine Learning with Adversaries : Byzantine Tolerant Gradient Descent. NIPS 2017.