紹介論文

Defending Against Saddle Point Attack in Byzantine-Robust Distributed Learning

Dong Yin, Yudong Chen, Kannan Ramchandran, Peter Bartlett

arXiv:1806.05358v3, 24 Jan 2019

Abstract

saddle points がある non-convex な損失関数を最小化する堅牢な分散学習を研究する. その上で, ビザンチン障害を想定する. ビザンチン障害を引き起こす Byzantine Worker は異常な行動や敵対的な行動をする. この問題設定においては, Byzantine Worker は saddle point の近くに偽の local minimum を作るかもしれない. そのような場合においても, saddle point や偽の local minimum を脱出でき, 少ないイテレーションで真の local minimum に収束する ByzantinePGD を示す.

1 Introduction

近年, 分散コンピューティングがますます重要になっている. 多くのアプリケーションでは, 計算を高速化するために, 大規模なデータを複数のマシンに分散して並列処理している. また, データソースは元々分散されており, プライバシーと計算効率を考慮して, データをマスターマシン (Server) に送信せずに, 並列処理している場合もある. その一例として Federated Learning[1] という手法がある.

標準的な分散学習では, 単一の Server と複数の Worker が存在し, Server は損失関数のパラメータの更新のみを行い, 各 Worker は自身のデータから損失関数の勾配の計算を行い, Server に送信する. 分散学習において, Worker と Server 間の通信はハードウェアやソフトウェアの誤動作・通信の遅延が起こる可能性は高い. さらに, 悪意ある攻撃, 協調的な攻撃や, 巧みなふるまいを受けるような問題設定を Byzantine setting[2] と呼ばれている. このような場合は, Byzantine Worker は任意な行動をとる. Byzantine setting を考慮した分散アルゴリズムの開発はますます重要になっている.

複雑な機械学習モデルでは, DNN などのように, non-convex な関数の local minimum を見つけることが要求される. このような問題での到達点の多くは, 実際には saddle point であり, いかなる local minimum からも遠く離れていることはよく知られている. このような場合では, 効率的に saddle point を脱出し, local minimum に到達するアルゴリズムが求められる. saddle point 脱出に関する研究は近年活発に行われている.

本論文での注目は, non-convex な損失関数とビザンチン障害によって, saddle point の脱出をより難しくしている. 特に, Worker が Server に送信するデータを Byzantine Worker が書き換えることによって, 真の local minimum から遠く離れた saddle point の近くに偽の local minimum を作成することもできる. このような戦略は, saddle point attack とも呼ばれ, 既存のアルゴリズムを無効にする.

1.1 Contributions

本論文では, Byzantine Perturbed Gradient Descent (ByzantinePGD) と呼ばれる saddle point と Byzantine Worker によって作成された偽の極小点を脱出し, non-convex な損失関数の近似的に極小点に収束することができるアルゴリズムを開発した. 我々の知る限りでは, ByzantinePGD は敵対的な環境下において, そのような保証を達成する最初のものである.

2 Related Work

我々のアルゴリズムは, 既存の saddle point を脱出するためのアルゴリズムに関係している. GD-based の収束性を保証したアルゴリズムの一般的な考え方は勾配が小さいときに, GD を行い, 摂動を足すという考えである (Table 1). しかし, 我々のアルゴリズムと以下の点で異なる.

- saddle point を脱出するためだけでなく, ランダム摂動は敵対的なエラーを防ぐ役割も担っている
- アルゴリズムで使用される摂動は, saddle point を脱出するのに, Byzantine worker による勾配の不確定さの影響を排除するためにより大きくする必要がある
- 損失関数そのものの値を使用しない (損失関数のパラメータを利用する) ; PGD や Neon+GD は損失関数の値を利用することを想定している
- saddle point の脱出の確率を高めるために, 複数の摂動を利用する

本研究では, median, trimmed mean, iterative filtering を用いる. 我々は, median, trimmed mean は [4] に従う. 一方, iterative filtering は [5] で提案されているが, 損失関数が strongly concex な場合のみしか考慮されておらず, 勾配に sub-exponential を仮定し, $d < O(\sqrt{mn})$ を仮定している. 我々は, 損失関数が non-convex な場合に適用し, d に制約を設けない. しかし, 勾配により強い仮定である sub-Gaussian を仮定する.

3 Problem Setup

表 1 Notation definition

Notation	Description
$m \in \mathbb{N}$	Worker 数
$\alpha \in (0, \frac{1}{2})$	敵対者の割合
$n \in \mathbb{N}$	各 Worker が取得するデータ数
$z \sim \mathcal{D} \in \mathcal{Z}$	分布 $\mathcal{D}$ に従うデータ点
$z_{i,j}$	Worker$i$ の j 番目のデータ点
$f(\boldsymbol{w}; z)$	学習のための損失関数
$\boldsymbol{w} \in \mathcal{W} \subseteq \mathbb{R}^d$	損失関数のパラメータ
$F(\boldsymbol{w}) \coloneqq \mathbb{E}_{z\sim\mathcal{D}}[f(\boldsymbol{w}; z)]$	期待誤差
$F_i(\boldsymbol{w}) \coloneqq \frac{1}{n}\sum_{j=1}^{n} f(\boldsymbol{w}; z_{i,j})$	Worker$i$ の経験誤差
$\mathcal{B}$	Byzantine Worker の集合 ($\lvert\mathcal{B}\rvert = \alpha m$)

表 1 に記法を示す. 各イテレーション毎に Server は全ての Worker にパラメータ $\boldsymbol{w}$ を送信し, 正常な Worker はそれぞれの経験誤差の勾配を送る. 一方で, Byzantine Worker は任意の敵対的な値を送信する. したがって, Worker$i$ は,

$$\hat{g}_i(\boldsymbol{w}) = \begin{cases} \nabla F_i(\boldsymbol{w}) & i \in [m] \setminus \mathcal{B}, \\ * & i \in \mathcal{B}, \end{cases} \tag{3.1}$$

ここで, $*$ は任意のベクトルを表す. Byzantine Worker は, 完全な知識を持っていると仮定され, Byzantine Worker 同士で結託することもできる. Byzantine Worker は, Server によって生成された乱数は予測できないという仮定のみしか存在しない.

我々は, $F(\boldsymbol{w})$ は non-convex であり, 近似的な local minimum を見つけることを目標とする. first-order stationary point (勾配が極めて小さい) は, Hessian が大きな負の固有値をもつ saddle point である可能性があるので, 必ずしも local minimum であるとは限らない. したがって, ほぼ半正定値であるような Hessian を持つような second-order stationary point を見つけることが目標となる.

Definition 1. (Second-order stationarity) $\|\nabla F(\tilde{\boldsymbol{w}})\|_2 \leq \epsilon_g$ と $\lambda_{min}\left(\nabla^2 F(\tilde{\boldsymbol{w}})\right) \geq -\epsilon_H$ を満たすならば, $\tilde{\boldsymbol{w}}$ は 2 回微分可能な関数 $F(\cdot)$ の (ϵ_g, ϵ_H)-second-order stationary point である. ただし, 固有値の最小値を $\lambda_{min}(\cdot)$ とする.

また, 連続最適化から標準的な概念を利用する.

Definition 2. (Smooth and Hessian-Lipschitz functions)$\sup_{\boldsymbol{w}\neq\boldsymbol{w}'} \frac{\|\nabla h(\boldsymbol{w})-\nabla h(\boldsymbol{w}')\|_2}{\|\boldsymbol{w}-\boldsymbol{w}'\|_2} \leq L$ を満たすならば, 関数 h は L-smooth である. また, $\sup_{\boldsymbol{w}\neq\boldsymbol{w}'} \frac{\|\nabla^2 h(\boldsymbol{w})-\nabla^2 h(\boldsymbol{w}')\|_2}{\|\boldsymbol{w}-\boldsymbol{w}'\|_2} \leq \rho$ を満たすならば, ρ-Hessian Lipschitz である.

本論文では, 期待誤差 $F(\cdot)$ に上記の性質を仮定する.

Assumption 1. F は $\mathcal{W}$ において, L_F-smooth であり, ρ_F-Hessian Lipschitz である.

4 Byzantine Perturbed Gradient Descent

Byzantine Worker 存在下の分散学習において, second-order stationary point に収束する ByzantinePGD について議論する. ByzantinePGD は Worker からの勾配を堅牢に集約し, Byzantine Worker の影響を防ぐために摂動を複数回足し合わせる. 我々は m 人の Worker から集められた勾配 $\{\hat{\boldsymbol{g}}_i(\boldsymbol{w})\}_{i=1}^m$ を堅牢に集約する GradAGG$\{\hat{\boldsymbol{g}}_i(\boldsymbol{w})\}_{i=1}^m$ を導入する. GradAGG は accuracy Δ で $\nabla F(\cdot)$ を推定できるとする. GradAGG の詳細については, section 5 で議論する.

Definition 3. (Inexact gradient oracle) 全ての $\boldsymbol{w} \in \mathcal{W}$ において, $\|\text{GradAGG}\{\hat{\boldsymbol{g}}_i(\boldsymbol{w})\}_{i=1}^m - \nabla F(\boldsymbol{w})\|_2 \leq \Delta$ を満たすならば, $\nabla F(\cdot)$ において GradAGG は Δ-inexact gradient oracle を与えると呼ぶ.

一般性を失わないために $\Delta \leq 1$ と仮定する.

しかし, 堅牢な集約規則を利用しても, 近似的に local minimum に収束することは保証されない. Byzantine Worker は偽の local minimum を作り出すかもしれない. ByzanitnePGD は saddle point と同様に偽の local minimum も脱出できるように設計されている.

4.1 Algorithm

ByzantinePGD のアルゴリズムを Algorithm 1 に示す. ただし, $[N] \coloneqq \{1, 2, \ldots, N\}$, 半径 r, 中心 $\boldsymbol{w}$, 次元 d の ℓ_2 ball を $\mathbb{B}_{\boldsymbol{w}}(r)$ とする.

まずは, $\|\hat{\boldsymbol{g}}(\boldsymbol{w})\|_2 \leq \epsilon$ を満たす $\tilde{\boldsymbol{w}}$ を得るまで, 通常のパラメータ更新を繰り返す. $\tilde{\boldsymbol{w}}$ は saddle point の可能性があるので, saddle point を脱出するために, Escape routine を設ける. Escape routine では, 各ラウンドで $\tilde{\boldsymbol{w}}$ にランダムな摂動を加えて, $\boldsymbol{w}_0'$ から最大で T_{th} 回のパラメータ更新を行う.

$$\boldsymbol{w}_t' = \boldsymbol{w}_{t-1}' - \eta \hat{\boldsymbol{g}}(\boldsymbol{w}_{t-1}'),\ t \leq T_{th}. \tag{4.1}$$

その間に, $\|\boldsymbol{w}_t' - \boldsymbol{w}_0'\|_2 \geq R$ を満たしたら, $\tilde{\boldsymbol{w}}$ は saddle point であり, 脱出したと主張する. もし Q ラウンドで $\tilde{\boldsymbol{w}}$ が改善しない場合, $\tilde{\boldsymbol{w}}$ は $F(\boldsymbol{w})$ は second-order stationary point であり, $\tilde{\boldsymbol{w}}$ を出力する.

4.2 Convergence Guarantees

ByzantinePGD の second-order stationary point への収束性についての理論保証を示す.

Observation 1. Assumpotion 1 と, GradAGG が Δ-inexact gradient oracle を与える ($\Delta \leq 1$) ならば, $\tilde{O}(\frac{1}{\Delta^2})$ イテレーション以内に, 以下を満たす $(O(\Delta), \tilde{O}(\Delta^{\frac{2}{5}}))$-second-order stationary point を出力する.

$$\|\nabla F(\tilde{\boldsymbol{w}})\|_2 \leq 4\Delta, \ \lambda_{min}(\nabla^2 F(\tilde{\boldsymbol{w}})) \geq -\tilde{O}(\Delta^{\frac{2}{5}}). \tag{4.2}$$

5 Robust Estimation of Gradients

5.3 Comparison and Optimality

Table 3 では, Δ に対して α, n, m, d の依存性について 3 つのアルゴリズムを比較している. $d = O(1)$ の場合, median と trimmed mean の方が優れているが, d が大きいときは, iterative filtering の方が好ましい.

Observation 1 より, Δ-inexact gradients the ByzantinePGD は $(O(\Delta), \tilde{O}(\Delta^{\frac{2}{5}}))$-second-order stationary point に収束する. この結果と, Table 3 を組み合わせると, ByzantinePGD の出力において明確な保証を得ることができる.

6 Conclusion

non-convex な損失関数に saddle point が存在するために大規模分散学習において生じるセキュリティ問題について考察した. non-convexity とビザンチン障害があると, saddle point を回避することがはるかに困難になる. 我々は Byzantine Worker 存在下でも, saddle point を脱出し, second-order stationary point に収束する ByzantinePGD を提案した.

References

[1] Jakub Konečn‘y, Brendan McMahan, and Daniel Ramage. Federated optimization: Distributed optimization beyond the datacenter. arXiv preprint arXiv:1511.03575, 2015.

[2] Leslie Lamport, Robert Shostak, and Marshall Pease. The Byzantine generals problem. ACM Transactions on Programming Languages and Systems (TOPLAS), 4(3):382 - 401, 1982.

[3] Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robust estimators in high dimensions without the computational intractability. In Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on, pages 655-664. IEEE, 2016.

[4] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In Proceedings of the 35th International Conference on Machine Learning, pages 5650-5659, 2018.

[5] Lili Su and Jiaming Xu. Securing distributed machine learning in high dimensions. arXiv preprint arXiv:1804.10140, 2018.