Federated Learning with Zeroth-Order Method Optimization

Yichuan Deng* Zhihang Li† Sridhar Mahadevan‡ Zhao Song§

Abstract

Federated Learning has emerged as a promising paradigm for training machine learning models across decentralized devices. This paper introduces a novel integration of zeroth-order optimization techniques into the Federated Learning paradigm. By incorporating zeroth-order methods, we enhance the computational efficiency. Traditional Federated Learning relies on gradient exchange during aggregation, leading to communication bottlenecks. Our approach eliminates gradient sharing, resulting in reduced communication overhead. The paper offers theoretical insights, algorithmic adaptations, and empirical validations, highlighting the enhanced efficiency and robustness of the proposed method. This innovative synergy between Federated Learning and zeroth-order optimization opens doors to scalable and efficient decentralized machine learning algorithms.

*ycdeng@cs.washington.edu. The University of Washington.
†lizhihangdll@gmail.com. Huazhong Agricultural University.
‡smahadev@adobe.com. Adobe Research
§zsong@adobe.com. Adobe Research.

Contents

1 Introduction

Federated learning [KMRR16, MMR+17] is a machine learning paradigm that enables model training across multiple decentralized edge devices or across multiple systems to keep data localized. Unlike traditional centralized training models, where all the data is sent to a central server, federated learning allows for the model to be trained at the source of the data itself. This approach has significant advantages in terms of privacy and security, as sensitive data never leaves the local device [KMRR16, MMR+17, GKN17, MRTZ17, CLK+18, BDF+18]. In federated learning, the global model is improved iteratively with local computations, and only model updates are transmitted and aggregated at a central server [KMRR16, MMR+17]. This decentralized approach is particularly beneficial in scenarios where data privacy is paramount, such as in healthcare [RHL+20, XGS+21, AAdCK+22] or Internet of things (See surveys [NDP+21, ITW+21, KSH+21]).

With the advantage of privacy-preserving property, communication is a critical bottleneck federated learning [KMY+16, MMR+17, LSTS20]. The exchange of model parameters or gradients $\nabla f(x)$ between decentralized devices and a central server can be resource-intensive and time-consuming. By addressing this bottleneck, it becomes possible to enhance the efficiency and scalability of federated learning algorithms, paving the way for more effective decentralized machine learning [LSTS20, LFTL20].

Zeroth-order optimization methods [Spa92, Spa98, FKM04, DJWW15, BP16, NS17] are optimization techniques that do not require access to the gradient of the objective function $f(x)$. Unlike first-order methods, which utilize the gradient $\nabla f(x)$, or second-order methods, which also take into account the Hessian matrix $\nabla^2 f(x)$, zeroth-order methods only require the function value $f(x)$ itself. This makes them particularly useful in scenarios where the gradient is either unavailable, expensive to compute, or unreliable. By employing techniques such as random sampling [Spa92, Spa98], where an estimate of the gradient is obtained as

$$\widehat{g} = \frac{f(x+\delta) - f(x)}{\delta} \text{ for a small } \delta,$$

zeroth-order methods can explore the search space and converge to a local or global minimum. The integration of zeroth-order methods in federated learning, as discussed in this paper, offers a novel approach to reduce the communication cost.

In this work, we focus our topic on an optimization problem inspired by Large Language models (LLMs). Recent years have seen remarkable advancements in natural language processing with the development of LLMs such as Transformer [VSP+17], GPT-1 [RNS+18], BERT [DCLT18], and GPT-4 by OpenAI [Ope23], which has notably outperformed its predecessors [BCE+23]. These LLMs excel in tasks like machine translation [HWL21], sentiment analysis [UAS+20], and creative writing [Cha22, Ope23]. A key feature is the *attention mechanism*, allowing focus on relevant text parts, implemented through soft weighting, and inspired by human attention [VSP+17, RNS+18, DCLT18, RWC+19, BMR+20, ZHDK23, AS23, BSZ23]. Inspired by attention computation, we focus our effort on the following optimization problem.

Definition 1.1 (Softmax Regression [DLS23])**.** *Let $A \in \mathbb{R}^{n \times d}$ and $b \in \mathbb{R}^n$ be a matrix and a vector, we define the objective function of softmax regression problem as*

$$\min_{x \in \mathbb{R}^d} \| \langle \exp(Ax), \mathbf{1}_n \rangle^{-1} \exp(Ax) - b \|_2^2.$$

1.1 Our Result

Based on the aforementioned zero-th order method, we obtained our result for the softmax regression prolem. Our main result is

Theorem 1.2 (Informal version of Theorem 5.4)**.** *Let $A_j \in \mathbb{R}^{n \times d}$, Let $b_j \in \mathbb{R}^n$ satisfy that $\|b_j\|_1 \leq 1$ for all $j \in [n]$. Let $R \geq 4$, $\|A_j\| \leq R$, $\|x\|_2 \leq R$, let $M := \exp(O(R^2 + \log n))$. Let $W = \mathrm{diag}(w)$, where $\min_i w_i^2 \geq \mu / \sigma_{\min}(A_j)$ for all $j \in [n]$, $|\mathcal{B}| = B$ let $\kappa(A) = \max_{j \in [n]} \kappa(A_j)$. Let*

$$\begin{aligned} & T \\ = & O(M(1 + d^{1.5}\kappa^2(A)/k) \cdot \mu^{-2} B^{-1} \log((L(x_0) - L^*)/\epsilon)). \end{aligned}$$

Let x^0 denote the init point of GD. Let $L^* = \min_x L(x)$. Then, there exists a federated learning algorithm, based on zero-th order method on multiple softmax loss function, converges to optimal with an additive error ϵ in T iterations.

1.2 Related Work

Optimization through Zero-th Order Methods. Zero-th Order (ZO) methods, essential for approximating gradients, have a rich history in optimization literature, dating back to the Kiefer-Wolfowitz algorithm in the 1950s [KW52]. These methods, including the simultaneous perturbation stochastic approximation (SPSA) [Spa87] and the Nelder-Mead simplex algorithm [NM65], estimate gradients using function evaluations without explicit derivatives. Recently, they have been applied to machine learning for handling nonsmooth objectives, constrained black-box models, and scenarios where gradients are costly [DJWW15, BG22, LRV+20], with key applications in adversarial attack generation [CZS+17], hyperparameter tuning [SLA12], and reinforcement learning [SHC+17]. Notable ZO algorithms include ZO gradient descent [NS17], ZO-SGD [GL13, MGN+23b, ZHL+23], and ZO Frank-Wolfe [BG22], some of which can optimize without direct gradient estimation [MGR18, GKK+19, Hin22]. Recent advancements include a memory-efficient variant of the SPSA algorithm [MGN+23b] and distributed fine-tuning with reduced bandwidth [ZHL+23], highlighting the crucial role of ZO optimization when first-order derivatives are infeasible, yet still enabling gradient-like algorithms. Recently [SLDL23] presents a federated zeroth-order optimization method, enhancing query and communication efficiency through trajectory-informed gradient surrogates and adaptive gradient correction, with validation in real-world applications.

Federated Learning. Federated learning is a decentralized machine learning approach that trains models across multiple edge devices, keeping data localized [KMRR16, MMR+17]. This method enhances privacy and security by training the model at the data source, ensuring sensitive data remains on the local device [KMRR16, MMR+17, GKN17, MRTZ17, CLK+18, BDF+18]. Particularly valuable in areas like healthcare and the Internet of Things, where data privacy is crucial [RHL+20, XGS+21, AAdCK+22, NDP+21, ITW+21, KSH+21], federated learning does face challenges in communication efficiency [KMY+16, MMR+17, LSTS20]. Recent work focuses on addressing this bottleneck to improve the scalability and effectiveness of federated learning algorithms [LSTS20, LFTL20, SLDL23].

Theoretical Understandings of LLMs. The explosion of large language models (LLMs) has led to extensive research in transformer theory. [SZKS21] explored how single-head attention modules learn in seq2seq translation, defining a property called knowing to translate individual word (KTIW). [GTLV22] empirically demonstrated transformers' ability to learn linear function classes, while [ASA+22] established a connection between transformers and traditional learning algorithms. [ZFB23] and [WZW23] further examined in-context learning abilities, with the latter viewing it through a Bayesian lens. Other studies have delved into the intrinsic structure of transformers, such as [ZPGA23], which discovered a parsing mechanism, and [PSZA23], which introduced the

concept of "skill location." The capabilities and limitations of transformers were empirically and analytically discussed by [SHT23], and mathematical insights into scaling up training data were provided by [AG23]. [BCE+23] conducted a comprehensive investigation of GPT-4, highlighting potential future research directions. In the realm of fast attention computation, techniques like locality sensitive hashing (LSH) and Kernel Density Estimation (KDE) have been employed to increase efficiency [KKL20, CLP+21, ZHDK23, PMXA23, MGN+23a]. LLM-related optimization has also been a focus, with novel algorithms and second-order optimizers being developed [CLMY21, RSM+23, LLH+23]. Over-parameterization in ReLU neural networks has been explored for convergence analysis [LL18, DZPS19, AZLS19, SYZ21, ALS+22]. Finally, LLM-inspired regression models have been analyzed with various activation functions, convergence analyses, and Lipschitz conditions [GMS23, LSZ23, DLS23, GSY23, GSX23, DLMS23], with the present work extending this analysis to a specific 2-layer regression problem with nonlinear units inspired by the transformer model.

2 Preliminary

In this section, we state preliminary for the whole paper. In Section 2.1, we state some tools for psd. In Section 2.2 we define some definitions to be used in this paper. In Section 2.3 we define some basic definition regarding to a function's properties. In Section 2.4 we state some tools from previous work. In Section 2.5 we privode the definition of Simultaneous Perturbation Stochastic Approximation(SPSA).

Algorithm 1 Federated Learning algorithm with using gradient

procedure FederatedLearningAlgorithm($\{L_i\}_{i\in[k]}$)
  Let T denote the total number of iterations
  All the k clients pick up the same initialization porint x_0
  for $t = 1 \to T$ **do**
    for $k = 1 \to k$ **do**
      Compute $g_i \leftarrow \nabla L_i(x_{t-1})$
      Send g_i to Server
    end for
    Server compute $g \leftarrow \sum_{i=1}^k g_i$
    Server send g to all k clients
    Each client updates its $x_t \leftarrow x_{t-1} - \eta \cdot g$
  end for
end procedure

2.1 Tools for PSD

Fact 2.1. *Let $x, y \in \mathbb{R}^d$, We have:*

- $xy^\top + yx^\top \preceq xx^\top + yy^\top$

Fact 2.2. *Let $\{\alpha_i\}_{i\in[n]} \subseteq \mathbb{R}^d$ be a set of vectors, then we have*

- *Part 1.* $a_i a_j^\top + a_j a_i^\top \preceq a_i a_i^\top + a_j a_j^\top$
- *Part 2.* $\sum_{i=1}^n \sum_{j>i}^n a_i a_j^\top + a_j a_i^\top \preceq (n-1) \sum_{i=1}^n a_i a_i^\top$

Algorithm 2 Federated Learning algorithm without using gradient

procedure ZEROFEDERATEDLEARNINGALGORITHM($\{L_i\}_{i\in[k]}$)
  Let T denote the total number of iterations
  All the k clients pick up the same initialization porint x_0
  for $t = 1 \to T$ **do**
    for $k = 1 \to k$ **do**
      Pick random ξ (say all clients share the same randomness so that they can pick same ξ)
      Compute evaluate $f_{1,i} \leftarrow L_i(x_{t-1} + \xi)$
      Compute evaluate $f_{2,i} \leftarrow L_i(x_{t-1} - \xi)$
      Send $f_{1,i}$ and $f_{2,i}$ to Server
    end for
    Server compute $g \leftarrow \sum_{i=1}^k (f_{1,i} - f_{2,i})/\xi$
    Server send g to all k clients
    Each client updates its $x_t \leftarrow x_{t-1} - \eta \cdot g$
  end for
end procedure

- *Part 3.* $\sum_{i=1}^n \sum_{j=1}^n a_i a_j^\top \preceq n \cdot \sum_{i=1}^n a_i a_i^\top$

Proof. **Proof of Part 1** It trivially follows from Fact 2.1

Proof of Part 2. We have

$$\begin{aligned}\sum_{i=1}^n \sum_{j>i}^n a_i a_j^\top + a_j a_i^\top \preceq & \sum_{i=1}^n \sum_{j>i}^n (a_i a_i^\top + a_j a_j^\top) \\ = & (n-1) \cdot \sum_{i=1}^n a_i a_i^\top\end{aligned}$$

where the first step follows from Part 1.

Proof of Part 3.

$$\begin{aligned}\sum_{i=1}^n \sum_{j=1}^n a_i a_j^\top = & \sum_{i=1}^n a_i a_i^\top + \sum_{i\neq j} a_i a_j^\top \\ = & \sum_{i=1}^n a_i a_i^\top + \sum_{i=1}^n \sum_{j>i}^n a_i a_j^\top + a_j a_i^\top \\ \preceq & n \sum_{i=1}^n a_i a_i^\top,\end{aligned}$$

where the first step follows from Fact 2.1, the second step follows from decomposing the second term, and the last step follows from Part 2. Thus we complete the proof. □

2.2 Basic Definitions

Definition 2.3 (Stable rank [CNW15])**.** *Let* $A \in \mathbb{R}^{n\times d}$

$$\mathrm{srank}(A) := \frac{\|A\|_F^2}{\|A\|^2}$$

to denote the stable rank of A.

Definition 2.4 (effective rank)**.** *Let $A \in \mathbb{R}^{d \times d}$, we use*

$$\mathrm{erank}(A) := \frac{\mathrm{tr}[A]}{\|A\|}$$

to denote the effective rank of A.

Definition 2.5 (Regularization Term)**.** *Let $A_j \in \mathbb{R}^{n \times d}$, $w \in \mathbb{R}^n$, $W = \mathrm{diag}(w)$. We define $L_{\mathrm{reg}} : \mathbb{R}^d \to \mathbb{R}$ as follows*

$$L_{\mathrm{reg},j}(x) := 0.5 \|W A_j x\|_2^2$$

Definition 2.6 (Our Softmax Loss Function)**.** *Let $x \in \mathbb{R}^d$, we define the softmax loss function as follows*

$$\begin{aligned} L_{\exp}(x) :=& \sum_{j=1}^n L_{\exp,j}(x) \\ L(x) :=& \sum_{j=1}^n L_{\exp,\mathrm{reg},j}(x) \end{aligned}$$

where

$$\begin{aligned} L_{\exp,j}(x) :=& 0.5 \| \langle \exp(A_j x), \mathbf{1}_n \rangle^{-1} \exp(A_j x) - b_j \|_2^2 \\ L_{\exp,\mathrm{reg},j}(x) :=& 0.5 \| \langle \exp(A_j x), \mathbf{1}_n \rangle^{-1} \exp(A_j x) - b_j \|_2^2 \\ & + 0.5 \|W A_j x\|_2^2 \end{aligned}$$

$A_j \in \mathbb{R}^{n \times d}$, $b_j \in \mathbb{R}^n$. For a certain batch $\mathcal{B} \in [n]$ of data points, we define

$$L_{\exp}(x; \mathcal{B}) := \sum_{j \in \mathcal{B}} L_{\exp,j}(x)$$

Definition 2.7. *We define $f_j(x)$ as follows*

$$f_j(x) := \langle \exp(A_j x), \mathbf{1}_n \rangle^{-1} \cdot \exp(A_j x).$$

Definition 2.8. *Let $b_j \in \mathbb{R}^n$.*
We define $c_j(x)$ as

$$c_j(x) := f_j(x) - b_j$$

Definition 2.9 ([DLS23])**.** *We define $B_j(x)$ as follows*

$$\begin{aligned} B_j(x) :=& \langle 3 f_j(x) - 2 b_j, f_j(x) \rangle f_j(x) f_j(x)^\top \\ & + (b_j \circ f_j(x)) f_j(x)^\top + f_j(x) (b_j \circ f_j(x))^\top \\ & + \langle f_j(x) - b_j, f_j(x) \rangle \cdot \mathrm{diag}(f_j(x)) \\ & + \mathrm{diag}((2 f_j(x) - b_j) \circ f_j(x)) \end{aligned}$$

Definition 2.10. *Given*

- *Let $f_j(x)$ be defined as Definition 2.7.*
- *Let $c_j(x)$ be defined as Definition 2.8.*

We define $G_j : \mathbb{R}^d \to \mathbb{R}^k$ as follows

$$G_j(x) := -\underbrace{f_j(x)}_{k\times 1}\underbrace{c_j(x)^\top}_{1\times k}\underbrace{f_j(x)}_{k\times 1} + \underbrace{\mathrm{diag}(f_j(x))}_{k\times k}\underbrace{c_j(x)}_{k\times 1}$$

For convenient, we define

Definition 2.11. *Given*

- *$f_j(x)$ follows from Definition 2.7.*
- *$b_j \in \mathbb{R}^k$*

We define $G_{j,1} : \mathbb{R}^d \to \mathbb{R}^k$ and $G_{j,2} : \mathbb{R}^d \to \mathbb{R}^k$

- *$G_{j,1} := f_j(x)(f_j(x) - b_j)^\top f_j(x)$*
- *$G_{j,2} := \mathrm{diag}(f_j(x))(f_j(x) - b_j)$*

Then it is obvious that $G_j(x) = -G_{j,1}(x) + G_{j,2}(x)$ (see Definition 2.10).

Definition 2.12 (Gradient Covariance, Definition 2.25 in [DLMS23])**.** *We say the covariance of GD gradient estimate over a dataset of N items is*

$$\Sigma(x) = N(\mathbb{E}[\nabla L(x)\nabla L(x)^\top] - \nabla L(x)\nabla L(x)^\top).$$

2.3 General Properties

Definition 2.13 (l-Smooth)**.** *We say a differentiable function $L(x) : \mathbb{R}^d \to \mathbb{R}$ is l-smooth if*

$$\|\nabla L(x) - \nabla L(y)\|_2 \leq l \cdot \|x - y\|_2, \quad \forall x, y \in \mathbb{R}^d.$$

Definition 2.14 (Convex)**.** *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a function, we say f is convex if for $\forall x, y \in \mathbb{R}^d$, we have*

$$f(x) \geq f(y) + \langle x - y, \nabla f(y)\rangle$$

Definition 2.15 (Stong Convexity)**.** *We say a continuously differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ is strongly convex if there exists a possitive number μ such that*

$$f(y) \geq f(x)\nabla f(x)^\top (y - x) + \frac{1}{2}\mu\|y - x\|_2^2, \quad \forall x, y \in \mathbb{R}^d.$$

Equivalently, if the function is twice differentiable, then

$$f(x) \text{ is } \mu - \text{strongly convex} \iff \nabla^2 f(x) \succeq \mu I.$$

Definition 2.16 (Polyak-Łojasiewicz Inequality)**.** *We say a function $L(x) : \mathbb{R}^d \to \mathbb{R}$ satisfies μ-Polyak-Łojasiewicz (PL) inequality if for all $x \in \mathbb{R}^d$, it holds that*

$$\frac{1}{2}\|\nabla L(x)\|^2 \geq \mu(L(x) - L^*),$$

where $L^* := \min_{x\in\mathbb{R}^d} L(x)$.

Definition 2.17 (Random Gaussian Matrix, Definition D.2 in [ZHL+23]). *We say $R \in \mathbb{R}^{b \times n}$ is a random Gaussian matrix if all its entries are sampled independently from $\mathcal{N}(0, 1/b)$.*

Problem 2.18. *Suppose there are k clients and the corresponding loss for client i ($i \in [k]$) is $L_i : \mathbb{R}^d \to R$, the problem we want to address in this paper is to find the minimal point for*

$$\min_{w \in \mathbb{R}^d} L(w) = \frac{1}{k} \sum_{i=1}^{k} L_i(w)$$

Assumption 2.19 (Assumption 3.1 in [ZHL+23]). *Assume there exists at least one minimizer for Problem 2.18, every loss function L_i is L-smooth and μ - strongly convex with parameter $\mu \geq 0$, which implies for $\forall x, y \in \mathbb{R}^d$, we have*

$$\frac{\mu}{2}\|y - x\|_2^2 \leq L_i(y) - L_i(x) + \langle y - x, \nabla L_i(x) \rangle \leq \frac{L}{2}\|y - x\|_2$$

2.4 Tools from Previous Work

Lemma 2.20 (GD version of Lemma 5 in [MGN+23b]). *Let $z \in \mathbb{R}^n$ with $z_i \sim \mathcal{N}(0, 1)$ i.i.d. Then it holds that*

$$\begin{aligned} & \mathbb{E}[\widehat{g}(x)\widehat{g}(x)^\top] \\ = & (1 + \frac{1}{n}) \cdot (\nabla L(x) L(x)^\top + \frac{1}{n}\Sigma(x)) \\ & + \frac{1}{n} I \cdot (\|\nabla L(x)\|^2 + \frac{1}{n} \operatorname{tr}(\Sigma(x))). \end{aligned}$$

Lemma 2.21 ([DLS23]). *Let $L_{\exp,\mathrm{reg},j}(x) \in \mathbb{R}^d$ follows from Definition 2.5, then we have*

$$\nabla^2 L_{\exp,\mathrm{reg},j}(x) \succeq \mu \cdot I_d$$

where $l > 0$ is a constant.

Lemma 2.22. *Let $B_j(x)$ be defined as Definition 2.9 and $L_{\exp}(x)$ be defined as Definition 2.6, then we have*

$$\nabla^2 L_{\exp}(x) = \sum_{j=1}^{n} A_j^\top B_j(x) A_j$$

Proof. It trivially follows from Lemma 5.10 of [DLS23]. □

Lemma 2.23 (Decomposition of gradient, [DLS23]). *Given*

- *$L_{\exp,j}(x)$ follows from Definition 2.6.*
- *$f_j(x)$ follows from Definition 2.7.*
- *$c_j(x)$ follows from Definition 2.8.*

Then it holds

- *Part 1.*

$$\nabla L_{\exp,j}(x) = A_j^\top \cdot G_j(x).$$

- *Part 2.*

$$\nabla L_{\exp}(x) = \sum_{j \in [n]} \nabla L_{\exp,j}(x)$$

We have the following existing lemma connecting strong-convexity and PL inequality.

Lemma 2.24 ([KNS16]). *If a function $L(x)$ is μ-strongly convex, then it is μ-PL.*

Fact 2.25 (Fact C.4 in [ZHL+23]). *Let $f : \mathbb{R}^d \to \mathbb{R}$ be an L-smooth and convex function, then for $\forall x, y \in \mathbb{R}^d$, we have*

$$f(y) - f(x) \geq \langle y - x, \nabla f(x) \rangle + \frac{1}{2L} \|\nabla f(y) - \nabla f(x)\|_2^2$$

Lemma 2.26 (Lemma D.13 in [ZHL+23]). *Let $R \in \mathbb{R}^{b \times n}$ denote a random Gaussian matrix which satisfies Definition 2.17. Then for any fixed vectors $h, g \in \mathbb{R}^n$, we have the following properties:*

$$\mathop{\mathbb{E}}_{\mathbb{R} \sim \Pi}[(g^\top R^\top R h)^2] \leq (g^\top h)^2 + \frac{3}{b} \|g\|_2^2 \cdot \|h\|_2^2$$

Lemma 2.27 (Lemma 5.2 in [DLS23]). *Let $f_j : \mathbb{R}^d \to \mathbb{R}$ follows from Definition 2.7, then for $\forall x \in \mathbb{R}^d$, it holds*

- $\|f_j(x)\|_2 \leq \|f_j(x)\|_1 \leq 1$.
- $0 \preceq f_j(x) f_j(x)^\top \preceq I_n$.
- *Let $b \in \mathbb{R}^d$,* $0 \preceq (b \circ f_j(x))(b \circ f_j(x))^\top \preceq \|b\|_\infty^2 f_j(x) f_j(x)^\top \preceq \|b\|_\infty^2 I_n$
- *Let $b \in \mathbb{R}^d$,* $\mathrm{diag}(b \circ b) \preceq \|b\|_\infty^2 I_n$
- $0 \preceq \mathrm{diag}(f_j(x)) \preceq \|f_j(x)\|_\infty I_n \preceq \|f_j(x)\|_2 I_n$.
- $0 \preceq \mathrm{diag}(f_j(x) \circ f_j(x)) \preceq \|f_j(x)\|_\infty^2 I_n \preceq \|f_j(x)\|_2 I_n$.

Fact 2.28 (Lemma 7.2 in [DLS23]). *If the following conditions hold*

- *Let $A \in \mathbb{R}^{n \times d}$*
- *Let $R \geq 4$*
- *Let $x, y \in \mathbb{R}^d$ satisfy $\|A(x - y)\|_\infty < 0.01$*
- *$\|A\| \leq R$*
- *Let $R_f := n^{1.5} \exp(5R^2)$*

We have

- *Part 0.* $\|\exp(Ax)\|_2 \leq \sqrt{n} \exp(R^2)$
- *Part 1.* $\|\exp(Ax) - \exp(Ay)\|_2 \leq 2\sqrt{n} R \exp(R^2) \cdot \|x - y\|_2$
- *Part 2.* $\|f_j(x) - f(y)\|_2 \leq R_f \cdot \|x - y\|_2$

Lemma 2.29 ([DLS23]). *If the following conditions holds*

- $\|A\| \leq R$
- $\|x\|_2 \leq R$
- *Let β be lower bound on $\langle \exp(Ax), \mathbf{1}_n \rangle$*

Then we have

$$\beta \geq \exp(-R^2)$$

Lemma 2.30 (Lemma 2 in [MGN+23b])**.** *Let L be defined as Definition 2.6, then for $\mathcal{B} \subset [n]$ we have*

$$\mathbb{E}[\|\widehat{g}(x, \mathcal{B})\|^2] = \frac{d+k-1}{k} \cdot \mathbb{E}[\|\nabla L(x, \mathcal{B})\|^2],$$

where k is the parameter for k-SPSA. We remark that the expectation is independent of $\mathcal{B}$.

2.5 Simultaneous Perturbation Stochastic Approximation (SPSA)

Definition 2.31 (Simultaneous Perturbation Stochastic Approximation (SPSA) [Spa92])**.** *Let $L(x)$ be a loss function. For a point $x_0 \in \mathbb{R}^d$, we define the Simultaneous Perturbation Stochastic Approximation (SPSA) of $L(x)$ on x_0 as a vector $\widehat{g}(x_0) \in \mathbb{R}^d$ such that*

$$\widehat{g}(x_0) := \frac{L(x_0 + \epsilon \cdot p) - L(x_0 - \epsilon \cdot p)}{2\epsilon} \cdot p, \quad \forall i \in [d],$$

where $p \in \mathbb{R}^d \sim \mathcal{N}(0, I_d)$ is the perturbation vector and $\epsilon > 0$ is the perturbation scale.

Remark 2.32 (k-SPSA)**.** *The k-SPSA gradient estimate averages $\widehat{g}(x)$ over k randomly sampled z.*

Lemma 2.33 ([Spa92])**.** *The gradient estimate $\widehat{g}(x)$ is almost unbiased, i.e.,*

$$\mathbb{E}[\widehat{g}(x)|x] = pp^\top \nabla L(x),$$

with probability of 1*.*

3 Analysis for Softmax Function

In this section, we provide analysis for the softmax loss function. In Section 3.1 we proved that the softmax loss function is strongly convex. In Section 3.2 we proved that the softmax loss function is smooth. In Section 3.3 we proved that $G_{j,1}$ is smooth. In Section 3.4 we proved that $G_{j,2}$ is smooth. In Section 3.5 we find the upper bound of the effecive rank of H by upper bounding the stable rank of $B(x)$. In Section 3.6 we state the inequality between stable rank and effective rank.

3.1 Softmax Loss is Strongly Convex

We have the following lemma

Lemma 3.1. *Let $L_{\exp,\mathrm{reg},j}(x)$ be defined as Definition 2.6, then there exists a parameter μ such that it is μ-strongly convex (Definition 2.15). And by Lemma 2.24, it is also μ-PL.*

3.2 Softmax Loss is Smooth

We have the following lemma

Lemma 3.2 (Lemma 3.1 in [DLMS23]). *Given*

- $A \in \mathbb{R}^{n \times d}$
- $R \geq 4$
- $x, y \in \mathbb{R}^d$ *satisfy* $\|A(x-y)\|_\infty < 0.01$
- $\|A\| \leq R$
- *Let* $R_f := n^{1.5} \exp(5R^2)$
- *Let* $W = \mathrm{diag}(w)$, *where* $w_i^2 \leq \frac{1}{\sigma_{\max}(A)}$.

Then the Softmax loss function (Definition 2.6) $L_j(x)$ *is* l*-smooth (Definition 2.13), where*

$$l = 8RR_f.$$

3.3 Smoothness for function $G_{j,1}$

Lemma 3.3 (Lemma 3.5 in [DLMS23]). *We define*

$$G_{j,1}(x) := f_j(x)(f_j(x) - b_j)^\top f_j(x).$$

Then we have

$$\|G_{j,1}(x) - G_{j,1}(y)\|_2 \leq 5R_f \cdot \|x - y\|_2.$$

3.4 Smoothness for function $G_{j,2}$

Lemma 3.4 (Lemma 3.6 in [DLMS23]). *We define*

$$G_{j,2}(x) := \mathrm{diag}(f_j(x))(f_j(x) - b_j).$$

Then we have

$$\|G_{j,2}(x) - G_{j,2}(y)\|_2 \leq 3R_f \cdot \|x - y\|_2.$$

3.5 Effective Bound for H

Lemma 3.5 (Upper Bound Stable Rank of B_j, Lemma 3.7 in [DLMS23]). *Let* B_j *be defined as in Lemma 2.22, then we have*

$$\mathrm{srank}(B_j) = (\frac{\|B_j\|_F}{\|B_j\|})^2 \leq 2d + 2,$$

where srank *is defined as Definition 2.3.*

3.6 The connection between effective rank and stable rank

The following lemma provide upper bound for the effective rank of H, in the term of stable rank of B.

Lemma 3.6 (Lemma 3.8 in [DLMS23]). *Let $A \in \mathbb{R}^{n \times d}, B \in \mathbb{R}^{n \times n}$ be two matrix. If the following conditions hold*

- $\|B\|_F / \|B\| \leq r$
- *Let $H = A^\top B A$*

Then,

$$\mathrm{erank}(H) \leq \mathrm{rank}(A) \cdot r \cdot \kappa^2(A),$$

where erank *is defined as Definition 2.4. Without loss of generality, we can assume $n \gg d$, then*

$$\mathrm{erank}(H) \leq dr \cdot \kappa^2(A).$$

4 Loss Analysis for Gradient Descent

In this section, we provide analysis for the loss in each iteration of the Gradient Descent. In Section 4.1, we define how we update the parameters in traditional SGD. In Sextion 4.2, we analyze the decrease of loss per iteration.

4.1 Gradient Step

Definition 4.1 (GD step). *The gradient descent step based on the zero-th order method is defined as*

$$x_{t+1} \leftarrow x_t - \eta \cdot \widehat{g}(x_t),$$

where $\widehat{g}(x_t)$ is defined as Definition 2.31.

4.2 Loss Decrease

Lemma 4.2 (Convergence Rate). *Let $x_{t+1} \leftarrow x_t - \eta \widehat{g}(x_t)$, where $\widehat{g}(x_t)$ is computed with respect to the batch $[n]$. Consider $L_{\exp}(x)$ as defined in Definition 2.6, then there exists a parameter*

$$\gamma = \frac{d^2 \cdot \sqrt{2d+2} \cdot \kappa^2(A) + d - 2}{k(d+2)} + 1$$

such that the expected loss decrease can be bounded as

$$\begin{aligned} & \mathbb{E}[L(x_{t+1})|x_t] - L(x_t) \\ \leq & -\eta \|\nabla L(x_t)\|^2 + \frac{1}{2}\eta^2 \ell \cdot \gamma \cdot \|\nabla L(x)\|^2 \end{aligned}$$

Proof. See Appendix B for detailed proof. □

Corollary 4.3 (Corollary 4.3 in [DLMS23]). *By Lemma 4.2 and Lemma 2.30, we choose $\eta = \eta_0$, where η_0 is used in traditional SGD. Then we have*

$$\begin{aligned} & \mathbb{E}[L(x_{t+1})|x_t] - L(x_t) \\ \leq & \frac{1}{\gamma} \cdot (-\eta_0 \|\nabla L(x_t)\|^2 + \frac{1}{2}\eta_0^2 \ell \cdot \|\nabla L(x)\|^2). \end{aligned}$$

where

$$\gamma = \frac{d^2 \cdot \sqrt{2d+2} \cdot \kappa^2(A) + d - 2}{k(d+2)} + 1$$

5 Convergence Analysis

In this section, we provide the analysis for convergence of our algorithm. During this section, we use $L^* := \min_{x \in \mathbb{R}^d} L(x)$ to denote the global minimum of $L(x)$. In Section 5.1, we upper bound the trace of covariance matrix under certain assumptions. In Section 5.2, we state an existing result with respect to the traditional SGD. In Section 5.3, we provide our main result, we show that our algorithm has convergence guarantee for softmax loss function.

Assumption 5.1. *Let $\epsilon_0 = 1/4$. We assume the following balanced distribution.*

1.

$$\begin{aligned} & \sum_{j \in [n]} A_j^\top G_j(x) G_j(x)^\top A_j \\ \approx & (1 \pm \epsilon_0) \sum_{j \in [n]} A_j^\top G_j(x) G_j(x)^\top A_j. \end{aligned}$$

2.

$$\begin{aligned} & \sum_{j_1 \neq j_2 \in [n]} A_{j_1}^\top G_{j_1}(x) G_{j_2}(x)^\top A_{j_2} \\ \approx & (1 \pm \epsilon_0) n(n-1) \cdot \frac{1}{n} (\sum_{j \in [n]} A_j^\top G_j(x) G_j(x)^\top A_j). \end{aligned}$$

5.1 Upper Bound Covariance

Lemma 5.2 (A gradient descent of Lemma 5.2 in [DLMS23]). *Let $\Sigma(x)$ be defined as Definition 2.12, If*

$$\begin{aligned} & \mathrm{tr}[\sum_{j \in [n]} A_j^\top G_j(x) G_j(x)^\top A_j] \\ \preceq & \epsilon_0^{-1} \frac{1}{n} \alpha (L(x) - L^*) \end{aligned} \tag{1}$$

Then we have

$$\mathrm{tr}[\Sigma(x)] \leq \alpha \cdot (L(x) - L^*),$$

for all $x \in \mathbb{R}^d$.

Proof. We have

$$\begin{aligned}
\operatorname{tr}[\Sigma(x)] \leq & |\operatorname{tr}[\Sigma(x)]| \\
= & |\operatorname{tr}[\nabla L(x) \nabla L(x)^\top - \mathbb{E}[\nabla L(x) \nabla L(x)^\top]]| \\
= & |\operatorname{tr}[\nabla L(x) \nabla L(x)^\top - \nabla L(x) \nabla L(x)^\top]| \\
= & 0 \\
\leq & \alpha \cdot (L(x) - L^*)
\end{aligned}$$

where step 1 follows from simple algebra, step 2 follows from the definition of $\Sigma(x)$, step 3 follows from simple algebra, step 4 follows from simple algebra, step 5 follows from $\alpha \cdot (L(x) - L^*) \geq 0$. □

5.2 Previous Results on SGD

We remark that previous work prove the following lemma for SGD setting. We modified it to gradient descent version.

Lemma 5.3 (A gradient descent version of Lemma 4 in [MGN+23b])**.** *Assume a loss function satisfies*

- *μ-PL (Definition 2.16);*
- *It holds that $\operatorname{tr}[\Sigma(x)] \leq \alpha \cdot (L(x) - L^*)$;*
- *l-smooth;*
- *Its Hessian H satisfies $\operatorname{erank}(H) \leq r$.*

Then after

$$O((\frac{l}{\mu} + \frac{l\alpha}{\mu^2 n}) \cdot \log \frac{L(x_0) - L^*}{\epsilon}))$$

iterations of GD (with the real gradient), it holds that

$$\mathbb{E}[L(x_t)] \leq L^* + \epsilon.$$

5.3 Global Convergence of the Zero-th Order Algorithm

In this section, we provide the following global convergence theorem.

Theorem 5.4 (Global convergence, formal version of Theorem 1.2)**.** *Given*

- *Let $A_j \in \mathbb{R}^{n \times d}$, $b_j \in \mathbb{R}^n$ satisfies $\|b_j\|_1 \leq 1$ for $\forall j \in [n]$*
- *Let $R \geq 4$, $\|A_j\| \leq R$, $\|x\|_2 \leq R$*
- *Let $W = \operatorname{diag}(w)$, where $\min_i w_i^2 \geq \mu / \sigma_{\min}(A_j)$ for all $j \in [n]$*
- *batch size $|\mathcal{D}| = n$*
- *Let let $\kappa(A) = \max_{j \in [n]} \kappa(A_j)$*
- *Let x_0 denote the initial point*

- *Let $L(x)$ be defined as Definition 2.6*
- *Let $L^* = \min_x L(x)$*
- *Let $M := \exp(O(R^2 + \log n))$*
- *Let*

$$t = O(M \cdot (1 + d^{1.5}\kappa^2(A)/k) \cdot \mu^{-2} n^{-1} \log((L(x_0) - L^*)/\epsilon)).$$

We perform GD algorithm based on zero-th order (Definition 4.1) gradient estimate on it. Then after t iterations, we have

$$\mathbb{E}[L(x_t)] \leq L^* + \epsilon.$$

Proof. See Appendix C for detailed proof. □

6 Conclusion

In this paper, we introduce a novel integration of zeroth-order optimization techniques into the Federated Learning paradigm. Unlike conventional methods that rely on the computation of gradients, denoted by $\nabla f(x)$, our approach leverages zeroth-order methods, which only necessitate the function value $f(x)$. This fundamental shift in methodology eliminates the need for gradient sharing among distributed nodes, resulting in a significant reduction in communication overhead. The advantage of this approach is twofold: it enhances computational efficiency by reducing the amount of data that must be transmitted, and it increases privacy by limiting the exposure of sensitive gradient information. By offering theoretical insights into the convergence and stability of the proposed method, algorithmic adaptations to suit various distributed environments, and empirical validations through extensive experiments on real-world datasets, we highlight the increased efficiency and robustness of the proposed method. The innovative synergy between Federated Learning and zeroth-order optimization not only provides a fresh perspective on decentralized optimization but also opens doors to scalable and efficient decentralized machine learning algorithms. This is particularly relevant in scenarios where communication constraints are paramount, such as edge computing, Internet of Things (IoT) devices, and remote sensing applications. The results of this work pave the way for future research and practical implementations that can harness the power of Federated Learning without the limitations imposed by traditional gradient-based methods.

References

[AAdCK+22] Rodolfo Stoffel Antunes, Cristiano André da Costa, Arne Küderle, Imrana Abdullahi Yari, and Björn Eskofier. Federated learning for healthcare: Systematic review and architecture proposal. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(4):1–23, 2022.

[AG23] Sanjeev Arora and Anirudh Goyal. A theory for emergence of complex skills in language models. *arXiv preprint arXiv:2307.15936*, 2023.

[ALS+22] Josh Alman, Jiehao Liang, Zhao Song, Ruizhe Zhang, and Danyang Zhuo. Bypass exponential time preprocessing: Fast neural network training via weight-data correlation preprocessing. *arXiv preprint arXiv:2211.14227*, 2022.

[AS23] Josh Alman and Zhao Song. Fast attention requires bounded entries. *arXiv preprint arXiv:2302.13214*, 2023.

[ASA+22] Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

[AZLS19] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. *Proceedings of the 36th International Conference on Machine Learning*, 2019.

[BCE+23] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

[BDF+18] Abhishek Bhowmick, John Duchi, Julien Freudiger, Gaurav Kapoor, and Ryan Rogers. Protection against reconstruction and its applications in private federated learning. *arXiv preprint arXiv:1812.00984*, 2018.

[BG22] Krishnakumar Balasubramanian and Saeed Ghadimi. Zeroth-order nonconvex stochastic optimization: Handling constraints, high dimensionality, and saddle points. *Foundations of Computational Mathematics*, pages 1–42, 2022.

[BMR+20] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[BP16] Francis Bach and Vianney Perchet. Highly-smooth zero-th order online optimization. In *Conference on Learning Theory*, pages 257–283. PMLR, 2016.

[BSZ23] Jan van den Brand, Zhao Song, and Tianyi Zhou. Algorithm and hardness for dynamic attention maintenance in large language models. *arXiv preprint arXiv:2304.02207*, 2023.

[Cha22] ChatGPT. Optimizing language models for dialogue. *OpenAI Blog*, November 2022.

[CLK+18] Nicholas Carlini, Chang Liu, Jernej Kos, Úlfar Erlingsson, and Dawn Song. The secret sharer: Measuring unintended neural network memorization & extracting secrets. *arXiv preprint arXiv:1802.08232*, 5, 2018.

[CLMY21] HanQin Cai, Yuchen Lou, Daniel Mckenzie, and Wotao Yin. A zeroth-order block coordinate descent algorithm for huge-scale black-box optimization. *arXiv preprint arXiv:2102.10707*, 2021.

[CLP+21] Beidi Chen, Zichang Liu, Binghui Peng, Zhaozhuo Xu, Jonathan Lingjie Li, Tri Dao, Zhao Song, Anshumali Shrivastava, and Christopher Re. Mongoose: A learnable lsh framework for efficient neural network training. In *International Conference on Learning Representations*, 2021.

[CNW15] Michael B Cohen, Jelani Nelson, and David P Woodruff. Optimal approximate matrix product in terms of stable rank. *arXiv preprint arXiv:1507.02268*, 2015.

[CZS+17] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*, pages 15–26, 2017.

[DCLT18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[DJWW15] John C Duchi, Michael I Jordan, Martin J Wainwright, and Andre Wibisono. Optimal rates for zero-order convex optimization: The power of two function evaluations. *IEEE Transactions on Information Theory*, 61(5):2788–2806, 2015.

[DLMS23] Yichuan Deng, Zhihang Li, Sridhar Mahadevan, and Zhao Song. Zero-th order algorithm for softmax attention optimization. *arXiv preprint arXiv:2307.08352*, 2023.

[DLS23] Yichuan Deng, Zhihang Li, and Zhao Song. Attention scheme inspired softmax regression. *arXiv preprint arXiv:2304.10411*, 2023.

[DZPS19] Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. *ICLR*, 2019.

[FKM04] Abraham D Flaxman, Adam Tauman Kalai, and H Brendan McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. *arXiv preprint cs/0408007*, 2004.

[GKK+19] Daniel Golovin, John Karro, Greg Kochanski, Chansoo Lee, Xingyou Song, and Qiuyi Zhang. Gradientless descent: High-dimensional zeroth-order optimization. *arXiv preprint arXiv:1911.06317*, 2019.

[GKN17] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

[GL13] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

[GMS23] Yeqi Gao, Sridhar Mahadevan, and Zhao Song. An over-parameterized exponential regression. *arXiv preprint arXiv:2303.16504*, 2023.

[GSX23] Yeqi Gao, Zhao Song, and Shenghao Xie. In-context learning for attention scheme: from single softmax regression to multiple softmax regression via a tensor trick. *arXiv preprint arXiv:2307.02419*, 2023.

[GSY23] Yeqi Gao, Zhao Song, and Junze Yin. An iterative algorithm for rescaled hyperbolic functions regression. *arXiv preprint arXiv:2305.00660*, 2023.

[GTLV22] Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. *arXiv preprint arXiv:2208.01066*, 2022.

[Hin22] Geoffrey Hinton. The forward-forward algorithm: Some preliminary investigations. *arXiv preprint arXiv:2212.13345*, 2022.

[HWL21] Weihua He, Yongyun Wu, and Xiaohua Li. Attention mechanism for neural machine translation: A survey. In *2021 IEEE 5th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, volume 5, pages 1485–1489. IEEE, 2021.

[ITW+21] Ahmed Imteaj, Urmish Thakker, Shiqiang Wang, Jian Li, and M Hadi Amini. A survey on federated learning for resource-constrained iot devices. *IEEE Internet of Things Journal*, 9(1):1–24, 2021.

[KKL20] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.

[KMRR16] Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

[KMY+16] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[KNS16] Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part I 16*, pages 795–811. Springer, 2016.

[KSH+21] Latif U Khan, Walid Saad, Zhu Han, Ekram Hossain, and Choong Seon Hong. Federated learning for internet of things: Recent advances, taxonomy, and open challenges. *IEEE Communications Surveys & Tutorials*, 23(3):1759–1799, 2021.

[KW52] Jack Kiefer and Jacob Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, pages 462–466, 1952.

[LFTL20] Li Li, Yuxi Fan, Mike Tse, and Kuo-Yi Lin. A review of applications in federated learning. *Computers & Industrial Engineering*, 149:106854, 2020.

[LL18] Yuanzhi Li and Yingyu Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. *NeurIPS*, 2018.

[LLH+23] Hong Liu, Zhiyuan Li, David Hall, Percy Liang, and Tengyu Ma. Sophia: A scalable stochastic second-order optimizer for language model pre-training. *arXiv preprint arXiv:2305.14342*, 2023.

[LRV+20] Sijia Liu, Parikshit Ram, Deepak Vijaykeerthy, Djallel Bouneffouf, Gregory Bramble, Horst Samulowitz, Dakuo Wang, Andrew Conn, and Alexander Gray. An admm based framework for automl pipeline configuration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4892–4899, 2020.

[LSTS20] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3):50–60, 2020.

[LSZ23] Zhihang Li, Zhao Song, and Tianyi Zhou. Solving regularized exp, cosh and sinh regression problems. *arXiv preprint, 2303.15725*, 2023.

[MGN+23a] Sadhika Malladi, Tianyu Gao, Eshaan Nichani, Alex Damian, Jason D. Lee, Danqi Chen, and Sanjeev Arora. Fine-tuning language models with just forward passes. *arXiv preprint arXiv:2305.17333*, 2023.

[MGN+23b] Sadhika Malladi, Tianyu Gao, Eshaan Nichani, Alex Damian, Jason D Lee, Danqi Chen, and Sanjeev Arora. Fine-tuning language models with just forward passes. *arXiv preprint arXiv:2305.17333*, 2023.

[MGR18] Horia Mania, Aurelia Guy, and Benjamin Recht. Simple random search of static linear policies is competitive for reinforcement learning. *Advances in Neural Information Processing Systems*, 31, 2018.

[MMR+17] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[MRTZ17] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. *arXiv preprint arXiv:1710.06963*, 2017.

[NDP+21] Dinh C Nguyen, Ming Ding, Pubudu N Pathirana, Aruna Seneviratne, Jun Li, and H Vincent Poor. Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 23(3):1622–1658, 2021.

[NM65] John A Nelder and Roger Mead. A simplex method for function minimization. *The computer journal*, 7(4):308–313, 1965.

[NS17] Yurii Nesterov and Vladimir Spokoiny. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17:527–566, 2017.

[Ope23] OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[PMXA23] Abhishek Panigrahi, Sadhika Malladi, Mengzhou Xia, and Sanjeev Arora. Trainable transformer in transformer. *arXiv preprint arXiv:2307.01189*, 2023.

[PSZA23] Abhishek Panigrahi, Nikunj Saunshi, Haoyu Zhao, and Sanjeev Arora. Task-specific skill localization in fine-tuned language models. *arXiv preprint arXiv:2302.06600*, 2023.

[RHL+20] Nicola Rieke, Jonny Hancox, Wenqi Li, Fausto Milletari, Holger R Roth, Shadi Albarqouni, Spyridon Bakas, Mathieu N Galtier, Bennett A Landman, Klaus Maier-Hein, et al. The future of digital health with federated learning. *NPJ digital medicine*, 3(1):119, 2020.

[RNS+18] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. ., 2018.

[RSM+23] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D.Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*, 2023.

[RWC+19] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[SHC+17] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

[SHT23] Clayton Sanford, Daniel Hsu, and Telgarsky. Representational strengths and limitations of transformers. *arXiv preprint arXiv:2306.02896*, 2023.

[SLA12] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

[SLDL23] Yao Shu, Xiaoqiang Lin, Zhongxiang Dai, and Bryan Kian Hsiang Low. Federated zeroth-order optimization using trajectory-informed surrogate gradients. *arXiv preprint arXiv:2308.04077*, 2023.

[Spa87] James C Spall. A stochastic approximation technique for generating maximum likelihood parameter estimates. In *1987 American control conference*, pages 1161–1167. IEEE, 1987.

[Spa92] James C Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE transactions on automatic control*, 37(3):332–341, 1992.

[Spa98] James C Spall. Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Transactions on aerospace and electronic systems*, 34(3):817–823, 1998.

[SYZ21] Zhao Song, Shuo Yang, and Ruizhe Zhang. Does preprocessing help training overparameterized neural networks? *35th Conference on Neural Information Processing Systems*, 2021.

[SZKS21] Charlie Snell, Ruiqi Zhong, Dan Klein, and Jacob Steinhardt. Approximating how single head attention learns. *arXiv preprint arXiv:2103.07601*, 2021.

[UAS+20] Mohd Usama, Belal Ahmad, Enmin Song, M Shamim Hossain, Mubarak Alrashoud, and Ghulam Muhammad. Attention-based sentiment analysis using convolutional and recurrent neural network. *Future Generation Computer Systems*, 113:571–578, 2020.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[WZW23] Xinyi Wang, Wanrong Zhu, and William Yang Wang. Large language models are implicitly topic models: Explaning and finding good demonstrations for in-context learning. *arXiv preprint arXiv:2301.11916*, 2023.

[XGS+21] Jie Xu, Benjamin S Glicksberg, Chang Su, Peter Walker, Jiang Bian, and Fei Wang. Federated learning for healthcare informatics. *Journal of Healthcare Informatics Research*, 5:1–19, 2021.

[ZFB23] Ruiqi Zhang, Spencer Frei, and Peter L Bartlett. Trained transformers learn linear models in-context. *arXiv preprint arXiv:2306.09927*, 2023.

[ZHDK23] Amir Zandieh, Insu Han, Majid Daliri, and Amin Karbasi. Kdeformer: Accelerating transformers via kernel density estimation. *arXiv preprint arXiv:2302.02451*, 2023.

[ZHL+23] Eric Zelikman, Qian Huang, Percy Liang, Nick Haber, and Noah D Goodman. Just one byte (per gradient): A note on low-bandwidth decentralized language model finetuning using shared randomness. *arXiv preprint arXiv:2306.10015*, 2023.

[ZPGA23] Haoyu Zhao, Abhishek Panigrahi, Rong Ge, and Sanjeev Arora. Do transformers parse while predicting the masked word? *arXiv preprint arXiv:2303.08117*, 2023.

Appendix

A Preliminaries

In this section, we provide the preliminaries to be used in the appendix. In Section A.1, we provide the definitions for the notations to be used in the appendix. In Section A.2, we state a tool for splitting the computation for computing the value of quadratic form. In Section A.3 we state some tools for matrix norm inequality.

A.1 Notations

In this paper, we use $\mathbb{R}$ to denote real numbers, $\mathbb{R}_{\geq 0}$ to denote non-negative real numbers.

Given vector $x \in \mathbb{R}^d$, we $b = \exp(x) \in \mathbb{R}^d$ to generate a vector such that $b_i = \exp(x_i)$ where $i \in [d]$

Given $x \in \mathbb{R}^n$, its ℓ_2-norm can be denote as $\|x\|_2 := (\sum_{i=1}^n x_i^2)^{1/2}$.

Given $A \in \mathbb{R}^{n \times k}$, its spectral norm can be denote as $\|A\|$, i.e.$\|A\| := \sup_{x \in \mathbb{R}^k} \|Ax\|_2 / \|x\|_2$.

Given A, its largest singular value is denoted as $\sigma_{\max}(A)$, its smallest singular value is denoted as $\sigma_{\min}(A)$.

Given $x \in \mathbb{R}^n$, we use $\|x\|_\infty$ to denote $\max_{i \in [n]} |x_i|$.

Given $x \in \mathbb{R}^n, y \in \mathbb{R}^n$, we use $c = x \circ y$ to generate a vector $c \in \mathbb{R}^n$ where $c_i = x_i y_i$ for $i \in [n]$.

Given $x \in \mathbb{R}^n$, we use $A = \mathrm{diag}(x) \in \mathbb{R}^{n \times n}$ to denote a diagonal matrix where $A_{i,i} = x_i$ for $i \in [n]$.

We use $a = \mathbf{1}_d \in \mathbb{R}^d$ to denote a vector such that $a_i = 1$ where $i \in [d]$

Given $A, B \in \mathbb{R}^{d \times d}$, we say $A \succeq B$ if $x^\top A x \geq x^\top B x$ for $\forall x \in \mathbb{R}^d$.

We define $\cosh(x) = \frac{1}{2}(\exp(x) + \exp(-x))$ and $\sinh(x) = \frac{1}{2}(\exp(x) - \exp(-x))$.

Given $A \in \mathbb{R}^{n \times d}$, we define the number of non zero entries of A to be $\mathrm{nnz}(A)$, i.e., $\mathrm{nnz}(A) := |\{(i,j) \in [n] \times [d] \mid A_{i,j} \neq 0\}|$

Given diagonal matrix $D \in \mathbb{R}^{n \times n}$, we say D is a k-sparse diagonal matrix where $k := |\{i \in [n] \mid D_{i,i} \neq 0\}|$.

Given function f, we use $\widetilde{O}(f)$ to denote $f \cdot \mathrm{poly}(\log f)$.

A.2 Basic Algebra

Fact A.1. • *Let $X \in \mathbb{R}^{k \times k}$, $a \in \mathbb{R}^k$, then*

$$a^\top X a = \sum_{i=1}^k \sum_{j=1}^k a_i X_{i,j} a_j = \sum_{i=1}^k a_i X_{i,i} a_i + \sum_{i \neq j} a_i X_{i,j} a_j.$$

A.3 Tools for Matrix Inequality

Fact A.2. *Let $A, B \in \mathbb{R}^{n \times d}$, then*

- $\|A\|_F \leq \sqrt{\mathrm{rank}(A)} \cdot \|A\|$
- $\mathrm{rank}(A + B) \leq \mathrm{rank}(A) + \mathrm{rank}(B)$
- $\|A^\top\| = \|A\|$
- $\|A\| \geq \|B\| - \|A - B\|$

- $\|A+B\| \leq \|A\| + \|B\|$
- $\|A \cdot B\| \leq \|A\| \cdot \|B\|$
- *Let* $a \in \mathbb{R}$, *if* $A \preceq a \cdot B$, *then* $\|A\| \leq a \cdot \|B\|$
- *Let* $a \in \mathbb{R}$, *then* $\|a \cdot A\| \leq |a| \cdot \|A\|$
- *Let* $x \in \mathbb{R}^d$, *we have* $\|Ax\|_2 \leq \|A\| \cdot \|x\|_2$.
- *Let* $x, y \in \mathbb{R}^d$, *then* $\|xy^\top\| \leq \|x\|_2 \|y\|_2$

B Proof of Lemma 4.2

In this section, we provide the proof for Lemma 4.2.

Lemma B.1 (Convergence Rate)**.** *Let* $x_{t+1} \leftarrow x_t - \eta \widehat{g}(x_t)$, *where* $\widehat{g}(x_t)$ *is computed with respect to the batch* $[n]$. *Consider* $L_{\exp}(x)$ *as defined in Definition 2.6, then there exists a parameter*

$$\gamma = \frac{d^2 \cdot \sqrt{2d+2} \cdot \kappa^2(A) + d - 2}{k(d+2)} + 1$$

such that the expected loss decrease can be bounded as

$$\begin{aligned} & \mathbb{E}[L(x_{t+1})|x_t] - L(x_t) \\ \leq & -\eta \|\nabla L(x_t)\|^2 + \frac{1}{2}\eta^2 \ell \cdot \gamma \cdot \|\nabla L(x)\|^2 \end{aligned}$$

Proof. By Taylor's theorem with remainder, we have that

$$\begin{aligned} & L(x_{t+1}) \qquad (2) \\ = & L(x_t) + \nabla L(x_t)^\top (x_{t+1} - x_t) \\ + & \int_0^1 \lambda (x_{t+1} - x_t)^\top \nabla^2 L(\lambda x_{t+1} + (1-\lambda) x_t)(x_{t+1} - x)^\top d\lambda. \qquad (3) \end{aligned}$$

Then by

$$\begin{aligned} \|x_{t+1} - x_t\| = & \eta \cdot \|\widehat{g}(x)\| \\ \leq & \eta \sqrt{d} \cdot \frac{1}{kn} \sum_{i=1}^k \sum_{j=1}^n |z_i^\top \nabla L_{\exp,j}(x)| \\ \leq & \eta d G_{\max}(x_t), \end{aligned}$$

where $G_{\max} := \max_{j \in [n]} \nabla L_{\exp,j}(x_t)$. The first step follows from the definition of GD step, the second step follows from the way we calculate $\widehat{g}$ (k-SPSA in Remark 2.32) , the third step follows from $|z_i^\top \nabla L_{\exp,j}(x)| \leq G_{\max}(x_t)$ and $\sqrt{d} \leq d$.

Thus we have

$$\|\lambda x_{t+1} + (1-\lambda) x_t - x_t\| \leq \eta d G_{\max}(x_t). \qquad (4)$$

this follows from simple algebra.

We define

$$H_\lambda(x_t) := \nabla^2 L(\lambda x_{t+1} + (1-\lambda)x_t). \tag{5}$$

Then we have

$$\begin{aligned}
& L(x_{t+1}) \\
\leq\ & L(x_t) + \nabla L(x_t)^\top (x_{t+1} - x_t) \\
& + (x_{t+1} - x_t)^\top H_\lambda(x_t)(x_{t+1} - x_t) \\
=\ & L(x_t) - \eta \nabla L(x_t)^\top \widehat{g}(x_t) + \frac{1}{2}\eta^2 \widehat{g}(x_t)^\top H_\lambda(x_t) \widehat{g}(x_t).
\end{aligned}$$

where step 1 follows from Eqs.(2), (4) and (5), step 2 follows from the way we update x_t.

We have

$$\begin{aligned}
& \mathbb{E}[L(x_{t+1})|x_t] \\
\leq\ & L(x_t) - \eta \|\nabla L(x_t)\|^2 + \frac{\eta^2}{2} \langle H_\lambda(x_t), \mathbb{E}[\widehat{g}(x)\widehat{g}(x)^\top] \rangle \\
=\ & L(x_t) - \eta \|\nabla L(x_t)\|^2 \\
+\ & \frac{\eta^2}{2} \cdot \frac{d}{k(k(d+2))} \cdot (\|\nabla L(x_t)\|^2 + \frac{1}{n} \operatorname{tr}[\Sigma(x_t))] \operatorname{tr}[H_\lambda(x_t)] \\
+\ & \frac{\eta^2}{2} (1 + \frac{d-2}{k(d+2)}) (\nabla L(x_t)^\top H_\lambda(x_t) \nabla L(x_t) + \frac{1}{n} \langle \Sigma(x_t), H_\lambda(x_t) \rangle).
\end{aligned}$$

where step 1 follows from taking conditional expectation with respect to x_t, step 2 follows from Lemma 2.20.

We have

- Part 1. $\|H_\lambda(x_t)\| \leq 8RR_f = l$, by Lemma 3.2;
- Part 2. $\operatorname{tr}[H_\lambda(x_t)]/\|H_\lambda(x_t)\| \leq d \cdot \sqrt{2d+2} \cdot \kappa^2(A) = r$, by Lemma 3.5 and Lemma 3.6.

Thus

$$\operatorname{tr}[H_\lambda(x_t)] \leq 8RR_f \cdot d \cdot \sqrt{2d+2} \cdot \kappa^2(A) = lr. \tag{6}$$

This follows from combining **Part 1** and **Part 2**.

Then we have

$$\begin{aligned}
& \mathbb{E}[L(x_{t+1})|x_t] \\
\leq\ & L(x_t) - \eta \|\nabla L(x_t)\|^2 \\
& + \frac{\eta^2 l}{2} \cdot (\frac{dr + d - 2}{k(d+2)} + 1) \cdot (\|\nabla L(x_t)\|^2 + \frac{1}{n} \operatorname{tr}[\Sigma(x_t)]) \\
=\ & L(x_t) - \eta \|\nabla L(x_t)\|^2 + \frac{\eta^2 l}{2} \cdot (\frac{dr + d - 2}{k(d+2)} + 1) \cdot \|\nabla L(x_t)\|^2.
\end{aligned}$$

where step 1 follows from Eq. (6), step 2 follows from definition of $\Sigma(x)$ (Definition 2.12).

Defining

$$\begin{aligned}
\gamma &= \frac{dr + d - 2}{k(d+2)} + 1 \\
&= \frac{d^2 \cdot \sqrt{2d+2} \cdot \kappa^2(A) + d - 2}{k(d+2)} + 1.
\end{aligned}$$

and we complete the proof. □

C Proof of Theorem 5.4

In this section, we provide the proof for Theorem 5.4.

Theorem C.1. *Given*

- *Let* $A_j \in \mathbb{R}^{n \times d}$, $b_j \in \mathbb{R}^n$ *satisfies* $\|b_j\|_1 \leq 1$ *for* $\forall j \in [n]$
- *Let* $R \geq 4$, $\|A_j\| \leq R$, $\|x\|_2 \leq R$
- *Let* $W = \mathrm{diag}(w)$, *where* $\min_i w_i^2 \geq \mu / \sigma_{\min}(A_j)$ *for all* $j \in [n]$
- *batch size* $|\mathcal{D}| = n$
- *Let let* $\kappa(A) = \max_{j \in [n]} \kappa(A_j)$
- *Let* x_0 *denote the initial point*
- *Let* $L(x)$ *be defined as Definition 2.6*
- *Let* $L^* = \min_x L(x)$
- *Let* $M := \exp(O(R^2 + \log n))$
- *Let*

$$\begin{aligned} & t \\ = & O(M \cdot (1 + d^{1.5}\kappa^2(A)/k) \cdot \mu^{-2} n^{-1} \log((L(x_0) - L^*)/\epsilon)). \end{aligned}$$

We perform GD algorithm based on zero-th order (Definition 4.1) gradient estimate on it. Then after t *iterations, we have*

$$\mathbb{E}[L(x_t)] \leq L^* + \epsilon.$$

Proof. Using Corollary 4.3, we obtain

$$\begin{aligned} & \mathbb{E}[L(x_{t+1})|x_t] - L(x_t) \\ \leq & \frac{1}{\gamma} \cdot [-\eta_0 \|\nabla L(x_t)\|^2 + \frac{1}{2}\eta_0^2 \ell \cdot \|\nabla L(x)\|^2, \end{aligned}$$

where η_0 is the learning rate used in traditional SGD. Note that

$$\|\nabla L(x_t)\|^2 \leq \|\nabla L(x_t)\|^2 + \frac{1}{n} \mathrm{tr}[\Sigma(x_t)].$$

By selecting $\eta_0 \leq \frac{1}{l}$, we have

$$\begin{aligned} & \mathbb{E}[L(x_{t+1})|x_t] - L(x_t) \\ \leq & \frac{1}{\gamma} \cdot (-\frac{\eta_0}{2} \|\nabla L(x_t)\|^2 + \frac{\eta_0^2 l}{2n} \mathrm{tr}[\Sigma(x_t)]). \end{aligned}$$

By Lemma 3.1 and Lemma 5.2, we have

$$\mathbb{E}[L(x_{t+1})|x_t] - L(x_t) \leq \frac{1}{\gamma}(-\eta_0 \mu + \frac{\eta_0^2 l \alpha}{2n}) \cdot (\mathbb{E}[L(x_t)] - L^*).$$

Thus by simple algebra, we obtain

$$\mathbb{E}[L(x_{t+1})] - L^* \leq (1 - \frac{1}{\gamma}(\eta_0\mu - \frac{\eta_0^2 l\alpha}{2n})) \cdot (\mathbb{E}[L(x_t)] - L^*).$$

Now by choosing $\eta_0 = \min\{\frac{1}{l}, \frac{\mu n}{l\alpha}\}$, we have

$$\mathbb{E}[L(x_{t+1})] - L^* \leq (1 - \frac{1}{\gamma} \cdot \min\{\frac{\mu}{2l}, \frac{\mu^2 n}{2l\alpha}\})(\mathbb{E}[L(x_t)] - L^*).$$

Now, to make $\mathbb{E}[L(x_t)] - L^* \leq \epsilon$, we need

$$t = \gamma \max\{\frac{2l}{\mu}, \frac{2l\alpha}{\mu^2 n}\} \log \frac{L(x_0) - L^*}{\epsilon}$$

iterations.

Plugging γ and l, we get

$$\begin{aligned}
& t \\
= & \ 16RR_f \cdot (\frac{d^2 \cdot \sqrt{2d+2} \cdot \kappa^2(A) + d - 2}{k(d+2)} + 1) \\
& \cdot \max\{\frac{1}{\mu}, \frac{\alpha}{\mu^2 n}\} \log \frac{L(x_0) - L^*}{\epsilon} \\
= & \ 16R\beta^{-2} n^{1.5} \exp(3R^2) \cdot (\frac{d^2 \cdot \sqrt{2d+2} \cdot \kappa^2(A) + d - 2}{k(d+2)} + 1) \\
& \cdot \max\{\frac{1}{\mu}, \frac{\alpha}{\mu^2 n}\} \log \frac{L(x_0) - L^*}{\epsilon} \\
= & \ 16Rn^{1.5} \exp(5R^2) \cdot (\frac{d^2 \cdot \sqrt{2d+2} \cdot \kappa^2(A) + d - 2}{k(d+2)} + 1) \\
& \cdot \max\{\frac{1}{\mu}, \frac{\alpha}{\mu^2 n}\} \log \frac{L(x_0) - L^*}{\epsilon} \\
= & \ O(n^{1.5} \exp(30R^2) \cdot (\frac{d^2 \cdot \sqrt{2d+2} \cdot \kappa^2(A) + d - 2}{k(d+2)} + 1) \\
& \cdot \mu^{-2} n^{-1} \log \frac{L(x_0) - L^*}{\epsilon}) \\
= & \ O(M \cdot (1 + d^{1.5} \cdot \kappa^2(A)/k) \cdot \mu^{-2} n^{-1} \log((L(x_0) - L^*)/\epsilon)),
\end{aligned}$$

where step 1 follows from plugging γ and l, step 2 follows from plugging R_f(Fact 2.28), step 3 follows from plugging β (Lemma 2.29), step 4 follows from $R \geq 4$ and the choosing α to be a large constant, step 5 follows from the definition of M.

Thus we complete the proof. □