ZO-AdaMU Optimizer: Adapting Perturbation by the Momentum and Uncertainty in Zeroth-order Optimization

Shuoran Jiang[1], Qingcai Chen[12*], Youcheng Pan[2*], Yang Xiang[2], Yukang Lin[1], Xiangping Wu[1], Chuanyi Liu[1], Xiaobao Song[3]

[1]Harbin Institute of Technology (Shenzhen), Shenzhen, China
[2]Peng Cheng Laboratory, Shenzhen, China
[3]Institute of Data Security, Harbin Institute of Technology QIANXIN
shuoran.chiang@gmail.com, qingcai.chen@hit.edu.cn, panyoucheng4@gmail.com, xiangy@pcl.ac.cn, 22S151097@stu.hit.edu.cn, wxpleduole@gmail.com, liuchuanyi@hit.edu.cn, songxiaobao@qianxin.com

Abstract

Lowering the memory requirement in full-parameter training on large models has become a hot research area. MeZO fine-tunes the large language models (LLMs) by just forward passes in a zeroth-order SGD optimizer (ZO-SGD), demonstrating excellent performance with the same GPU memory usage as inference. However, the simulated perturbation stochastic approximation for gradient estimate in MeZO leads to severe oscillations and incurs a substantial time overhead. Moreover, without momentum regularization, MeZO shows severe over-fitting problems. Lastly, the perturbation-irrelevant momentum on ZO-SGD does not improve the convergence rate. This study proposes ZO-AdaMU to resolve the above problems by adapting the simulated perturbation with momentum in its stochastic approximation. Unlike existing adaptive momentum methods, we relocate momentum on simulated perturbation in stochastic gradient approximation. Our convergence analysis and experiments prove this is a better way to improve convergence stability and rate in ZO-SGD. Extensive experiments demonstrate that ZO-AdaMU yields better generalization for LLMs fine-tuning across various NLP tasks than MeZO and its momentum variants.

Introduction

Large-scale models demonstrate remarkable abilities such as emergence and grokking (Wei et al. 2022), especially the large language models (LLMs) show excellent in-context learning (ICL) abilities and revolutionized the dominant methodology in various natural language processing (NLP) tasks. However, full-parameter fine-tuning with billions of parameters raises the bar for most NLP researches (Lv et al. 2023). Malladi et al. (2023a) experimentally proved that back-propagation optimization necessitates approximately 12 times of memory cost of forward inference. full-parameter fine-tuning a 6.7 billion (B) OPT (Zhang et al. 2022) with Adam (Kingma and Ba 2015) requires at least 3$\times$ A100 GPUs (240GB memory).

Therefore, the memory-efficient fine-tuning method has become an important research topic in the large-scale model era. Some approaches have been proposed, such as ICL does not need optimization (Sun et al. 2023a) and quickly adapts LLMs to specific use cases through demonstrations before test examples. However, a few demonstrations can only cover some possible case typos due to the limited maximum sequence length for LLMs inputs. The parameter-efficient fine-tuning methods (PEFT) (Fu et al. 2023), e.g., LoRA (Hu et al. 2021), ZeRA (Rajbhandari et al. 2020), update only a fraction of the model parameters. Even though these methods can tune LLMs with low memory and computation resources, there are more practical solutions to fully use the emergent ability as full-parameters fine-tuning (Ding et al. 2022; Sun et al. 2023b).

The memory-efficient full-parameter fine-tuning method is a better way to exploit the deployment of LLMs on limited resources. The low-memory optimization (LOMO) gives up gradient storage in stochastic gradient descent (SGD) (Shamir and Zhang 2013), but computes gradients at each training step. Zeroth-order optimization (ZO) relies solely on forward passes to estimate gradients and update parameters, costing the same memory size as inference. Malladi et al. (2023a) proposed the memory-efficient zeroth-order optimizer (MeZO) to fine-tune LLMs with just forward passes, and they achieved excellent performance on various NLP tasks. The surprising performance of MeZO is attributed to the small local effective rank of Hession parameter matrices in pre-trained deep neural networks (Papyan 2018, 2020; Ghorbani, Krishnan, and Xiao 2019; Yao et al. 2020; Wu et al. 2020; Sagun et al. 2017). MeZO computes gradients from two forward passes with random perturbations. Therefore, MeZO greatly reduces the latency between GPU calculations compared to LOMO. However, the simulated perturbation stochastic approximation for the gradient in MeZO is non-smoothness in consecutive training steps. Without the regularization from momentum, like in back-propagation optimization methods, MeZO causes the over-fitting problem. ZO-AdaMM (Chen et al. 2019) first efforts to integrate adaptive momentum methods with ZO optimization, but the perturbation-irrelevant momentum requires longer training steps to convergence.

Motivated by these challenges, we propose the ZO-AdaMU optimizer, in which we first effort to introduce adaptive momentum and uncertainty on simulated perturbation to approximate gradients. Specifically, we relocate the momentum from gradient to simulated perturbation,

*Corresponding Authors.

which aims to improve the smoothness between gradient approximation and actual parameter moving. In addition, the simulated perturbation stochastic approximation (SPSA) (Maryak and Chin 2001) includes two parts sampled from momentum-centered and zero-centered Gaussian distribution. The momentum-centered Gaussian in SPSA includes a uncertainty and its value is set by a simulated annealing function. With the introduction of adaptive momentum and uncertainty, ZO-AdaMU demonstrates high stability and faster convergence speed. Our convergence analysis and comprehensive experiments prove these for stable convergence rates and better global convergence. As some gradients in ZO-AdaMU inevitably deviate from the prediction, we propose a second-order momentum approximation to improve the convergence rate further. The details of ZO-AdaMU are summarized in Algorithm 1.

Contributions of our paper can be summarized as follows:

- Motivated by the problems of oscillation in MeZO and perturbation-irrelevant momentum in ZO-AdaMM, we first efforts to explore a way to adapt momentum and uncertainty in the zeroth-order oracle, called ZO-AdaMU.
- Our theoretical analysis proved that momentum with uncertainty and its relocation to simulated perturbation improve the convergence rate. Meanwhile, these also contribute to a better local optimal point than a well-tuned MeZO counterpart.
- Our comprehensive experiments evaluate a variety of NLP tasks and demonstrate that ZO-AdaMU shows a faster convergence rate and better generalization capability than MeZO, LOMO, and Adam fine-tuned full-parameter LLMs.

Preliminaries

Zeroth-order optimization (ZO) is a gradient-free method, and it estimates the gradient via the simulated perturbations stochastic approximation (SPSA). This section briefly introduces the multi-point estimate version as in MeZO. In addition, the momentum-based regularization methods for ZO optimization are also described. cause-effect reasoning (COPA),

Zeroth-Order Optimization

Zeroth-order (ZO) optimization has long been studied in the context of convex and strongly convex objectives. A classical ZO gradient estimator is a simultaneous perturbation stochastic approximation (SPSA) (Maryak and Chin 2001), and it can replace the gradient calculation in stochastic gradient descent (ZO-SGD) (Ghadimi and Lan 2013).

Consider a labelled dataset $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i \in \|\mathcal{D}\|}$, a minibatch $\mathcal{B} \subset \mathcal{D}$ of size B, and let $\mathcal{L}(\boldsymbol{\theta}; \mathcal{B})$ represents the loss on the minibatch for a model with parameters $\boldsymbol{\theta} \in \mathbb{R}^d$. The ZO gradient estimate via SPSA is defined as,

$$\begin{aligned} \widehat{\nabla}\mathcal{L}(\boldsymbol{\theta}; \mathcal{B}) &= \frac{\mathcal{L}(\boldsymbol{\theta} + \epsilon \boldsymbol{z}; \mathcal{B}) - \mathcal{L}(\boldsymbol{\theta} - \epsilon \boldsymbol{z}; \mathcal{B})}{2\epsilon} \boldsymbol{z} \\ &\approx \boldsymbol{z}\boldsymbol{z}^\top \nabla \mathcal{L}(\boldsymbol{\theta}; \mathcal{B}) \end{aligned} \tag{1}$$

where $\boldsymbol{z} \in \mathbb{R}^d$ with $\boldsymbol{z} \sim \mathcal{N}(0, \boldsymbol{I})$, $\boldsymbol{I} \in \mathbb{R}^{\|\boldsymbol{\theta}\|}$ and ϵ is the perturbation scale.

SPSA requires only two forward passes through model $\boldsymbol{\theta}$ to estimate the gradient. SPSA estimated gradients can be used to replace the gradient computation in any back-propagation optimizer such as SGD.

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \widehat{\nabla}\mathcal{L}(\boldsymbol{\theta}; \mathcal{B}_t) \tag{2}$$

where $\mathcal{B}_t$ denotes the monibatch at step t and $\widehat{\nabla}$ is the SPSA estimated gradient.

Memory-efficient ZO-SGD (MeZO)

MeZO (Malladi et al. 2023a) is an in-place implementation of ZO-SGD, and it further reduces memory requirements with the same memory usage as the inference. Specifically, MeZO keeps the random seed s for all random vector $\boldsymbol{z}$ sampling to generate each perturbation $\boldsymbol{z}$ at each step.

$$\begin{aligned} &\mathcal{B} \in \mathcal{D}, \qquad s \leftarrow \texttt{rand}() \\ &\boldsymbol{\theta} \leftarrow \texttt{PerturbParameters}(\boldsymbol{\theta}, \epsilon, s) \\ &\ell_+ \leftarrow \mathcal{L}(\boldsymbol{\theta}, \mathcal{B}) \\ &\boldsymbol{\theta} \leftarrow \texttt{PerturbParameters}(\boldsymbol{\theta}, -2\epsilon, s) \\ &\ell_- \leftarrow \mathcal{L}(\boldsymbol{\theta}, \mathcal{B}) \\ &\boldsymbol{\theta} \leftarrow \texttt{PerturbParameters}(\boldsymbol{\theta}, \epsilon, s) \\ &\texttt{grad} \leftarrow (\ell_+ - \ell_-) / (2\epsilon) \\ &\boldsymbol{z} \sim \mathcal{N}(0, 1) \quad \texttt{with seed } s \\ &\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta * \texttt{grad} * \boldsymbol{z} \end{aligned} \tag{3}$$

where $s \leftarrow \texttt{rand}()$ samples a random seed keeping same in one gradient updating step, η is the learning rate and ϵ is the perturbation scale.

Adaptive Momentum Method for ZO

ZO-AdaMM (Chen et al. 2019) first efforts to integrate adaptive momentum methods with ZO-SGD, and it shows theoretically convergence guarantees for both convex and non-convex constrained optimization. The method is inspired by the Adam optimization algorithm and extends its capabilities to handle scenarios where gradient information is unavailable. Zo-AdaMM leverages both the zeroth-order moments, which capture the function's behavior, and the historical information of previous iterations to dynamically adjust the step size and momentum for optimization.

This method has the potential to advance optimization techniques in scenarios where gradient information is not available, opening new avenues for solving complex real-world optimization problems. However, it suffers a slow-down factor $\mathcal{O}(\sqrt{d})$ compared with first-order Adam.

Methodology

In this section, we propose the ZO-AdaMU optimizer by adapting perturbation with the momentum and uncertainty in the zeroth-order oracle. Unlike the post-hoc momentum estimate in back-propagation optimization methods, ZO-AdaMU adapts the simulated perturbation with momentum and introduces the adaptive uncertainties in stochastic approximation. In addition, we propose a simulated annealing

mechanism on uncertainty in SPSA and smoothing parameters to balance the weight of momentum in gradient approximation. We theoretically analyze that ZO-AdaMU has a faster convergence rate and better global convergence.

Adapting Momentum by Momentum and Uncertainty in SPSA

The simulated perturbation in ZO-AdaMU is computed from two parts of a momentum-centered and a zero-centered Gaussian distribution:

$$\begin{aligned} \dot{\boldsymbol{z}}_{t+1} &\sim \mathcal{N}\left(0, \sqrt{\alpha_{t+1}}\right) \\ \ddot{\boldsymbol{z}}_{t+1} &\sim \mathcal{N}\left(\boldsymbol{m}_t, \sqrt{1-\alpha_{t+1}}\right) \\ \boldsymbol{m}_{t+1} &= \beta_1 \dot{\boldsymbol{z}}_{t+1} + (1-\beta_1)\ddot{\boldsymbol{z}}_{t+1} \\ &\text{s.t.} \quad 0 \leqslant \alpha_{t+1} \leqslant 1, \quad 0 \leqslant \beta_1 \leqslant 1 \end{aligned} \tag{4}$$

where $\boldsymbol{m}_t$ and α_{t+1} denote the momentum of history perturbations and the adaptive uncertainty, these variables also imply the mean and variance for momentum-centered Gaussian distribution. Consistently, 0 and $1-\alpha_{t+1}$ represent the mean and variance for zero-centered Gaussian distribution. $\dot{\boldsymbol{z}}_{t+1}$ and $\ddot{\boldsymbol{z}}_{t+1}$ represent the momentum with uncertainty and purely stochastic perturbation in SPSA. The hyper-parameter β_{t+1}^1 is the smoothing parameter, and it is also adaptive with a simulated annealing function.

In this way, the gradient on the minibatch $\mathcal{B}_t \in \mathcal{D}$ for a given model $f(\boldsymbol{\theta})$ is estimated by,

$$\widehat{\nabla}\mathcal{L}\left(\boldsymbol{\theta}; \mathcal{B}_t\right) = \frac{\mathcal{L}\left(\boldsymbol{\theta} + \epsilon \boldsymbol{m}_t; \mathcal{B}_t\right) - \mathcal{L}\left(\boldsymbol{\theta} - \epsilon \boldsymbol{m}_t; \mathcal{B}_t\right)}{2\epsilon} \boldsymbol{m}_t \tag{5}$$

ZO-AdaMU also mimics the exponential moving average (EMA) on the square of the gradient (Zhuang et al. 2020) to regularize step size, and proposes a substitute defined as follows,

$$\begin{aligned} \boldsymbol{v}_{t+1} &= \beta_{t+1}^2 \dot{\boldsymbol{z}}_{t+1}^2 + (1-\beta_{t+1}^2)\ddot{\boldsymbol{z}}_{t+1}^2 \\ \boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t - \eta \frac{\widehat{\nabla}\mathcal{L}(\boldsymbol{\theta}, \mathcal{B}_t)}{\sqrt{\boldsymbol{v}_{t+1}^2 + \sigma}} \end{aligned} \tag{6}$$

where β_{t+1}^2 is an adaptive smoothing parameter and σ is a small noise and typically set as 10^{-8}.

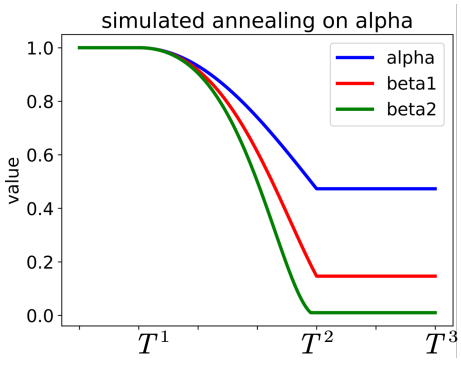

Figure 1: The simulated annealing on α, β_1 and β_3.

The big change from traditional adaptive momentum methods is that the smoothing parameters β_t^1, β_t^2 and uncertainty α_t are adaptive in a simulated annealing way as follows,

$$\texttt{Anneal}\,(t) = \begin{cases} 1, & t \in [1, T^1) \\ 0.5 + 0.5\cos\left(\pi \frac{(t^3 - T^1)}{T^3 - t\varphi \frac{T^3 - T^2}{T^2 - T^1}}\right), & t \in [T^1, T^2) \\ 0.9, & t \in [T^2, T^3) \end{cases} \tag{7}$$

where $\varphi = 1$ for α, $\varphi = 0.1$ for β_1 and $\varphi = 1.5$ for β_2. $t \in [1, T^1)$ is the warm-up process to estimate a major optimization orientation in SPSA without any influence of momentum. The second $t \in [T^1, T^2)$ process accelerates the optimization by momentum-based stochastic approximation, the uncertainty α on momentum gradually increases until 0.5 as shown in Figure 1. The last $t \in [T^2, T^3)$ process fixes $\alpha = 0.5$, $\beta_t^{(1)} = 0.9$ and $\beta_t^{(2)} = 0.01$ to find a global convergence. The simulated annealing is plotted in Figure 1.

The proposed ZO-AdaMU is summarized in Algorithm 1.

Algorithm 1: ZO-AdaMU Optimizer

Input: parameters $\boldsymbol{\theta} \in \mathbb{R}^d$, loss $\mathcal{L} : \mathbb{R}^d \to \mathbb{R}$, step budget T^1, T^2, T^3, perturbation scale ϵ, small number $\sigma = 10^{-8}$, batch size B, momentum uncertainty α, learning rate η, EMA of perturbation $\boldsymbol{m}$ and $\boldsymbol{v}$ EMA on its square.
for $t = 1, \cdots, T$ **do**
  Sample batch $\mathcal{B}_t \subset \mathcal{D}$ and random seed s
  $\boldsymbol{\theta} \leftarrow \texttt{Perturb}(\boldsymbol{\theta}, \boldsymbol{m}^{(t)}\epsilon, s, t)$, $\ell_+ \leftarrow \mathcal{L}(\boldsymbol{\theta}; \mathcal{B}_t)$
  $\boldsymbol{\theta} \leftarrow \texttt{Perturb}(\boldsymbol{\theta}, \boldsymbol{m}^{(t)}, -2\epsilon, s, t)$, $\ell_- \leftarrow \mathcal{L}(\boldsymbol{\theta}; \mathcal{B}_t)$
  $\boldsymbol{\theta} \leftarrow \texttt{Perturb}(\boldsymbol{\theta}, \boldsymbol{m}^{(t)}, \epsilon, s, t)$, $g_t \leftarrow (\ell_+ - \ell_-)/(2\epsilon)$
  Reset random seed s
  for $\theta_i \in \boldsymbol{\theta}$ **do**
    $\alpha, \beta_1^{(t)}, \beta_2^{(t)} = \texttt{Anneal}(t)$
    $\dot{\boldsymbol{z}} \sim \mathcal{N}(0, \sqrt{\alpha})$, $\ddot{\boldsymbol{z}} \sim \mathcal{N}(m_i^{(t-1)}, \sqrt{1-\alpha})$
    $m_i^{(t)} \leftarrow \beta_1^{(t)} \cdot \dot{\boldsymbol{z}} + (1-\beta_1^{(t)}) \cdot \ddot{\boldsymbol{z}}$
    $v_i = \beta_2^{(t)} \cdot \dot{z}^2 + (1-\beta_2^{(t)}) \cdot \ddot{z}^2$
    $\theta_i \leftarrow \theta_i - \eta_t \cdot \frac{g_t}{\sqrt{\boldsymbol{v}+\sigma}} \cdot m_i^{(t)}$
  end for
end for
Subroutine $\texttt{Perturb}(\boldsymbol{\theta}, \boldsymbol{m}, \epsilon, s, t)$
  Reset random seed s
  $\alpha^{(t)}, \beta_1^{(t)} \leftarrow \texttt{Anneal}(t)$
  $\dot{\boldsymbol{z}} \sim \mathcal{N}(0, \boldsymbol{I} * \sqrt{\alpha^{(t)}})$, $\ddot{\boldsymbol{z}} \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{I} * \sqrt{1-\alpha^{(t)}})$
  $\boldsymbol{\theta} \leftarrow \epsilon * (\beta_1^{(t)} \cdot \dot{\boldsymbol{z}} + (1-\beta_1^{(t)}) \cdot \ddot{\boldsymbol{z}})$
Subroutine $\texttt{Anneal}(t, T^1, T^2, T^3)$
  $\alpha \leftarrow \texttt{Anneal}(t)$ `# refer to Eq. (7)`

Convergence Analysis

We give the theoretical analysis about why ZO-AdaMU has higher convergence rate and better global convergence. We follow the convergence analysis in MeZO and pay more attention to why adapting perturbation with momentum and

Task Task type	SST-2	RTE	CB	BoolQ	WSC classification	WIC	MultiRC	COPA	ReCoRD multiple choice	SQuAD	DROP generation
Zero-shot	58.8	59.6	46.4	59.0	38.5	55.0	46.9	80.0	81.2	46.2	14.6
In-context	87.0	62.1	57.1	66.9	39.4	50.5	53.1	87.0	82.5	75.9	29.6
linear probing	93.4	68.6	67.9	59.3	63.5	60.2	63.5	55.0	27.1	3.7	11.1
MeZO	91.4	66.1	67.9	67.6	63.5	**61.1**	60.1	88.0	81.7	84.7	30.9
MeZO (LoRA)	89.6	67.9	66.1	73.8	**64.4**	59.7	61.5	87.0	81.4	83.8	31.4
MeZO (prefix)	90.7	70.8	69.6	73.1	57.7	59.9	**63.7**	84.0	81.2	84.2	28.9
ZO-AdaMU (2×)	**92.1**	**72.9**	67.9	73.0	61.5	60.7	63.0	**89.0**	83.0	82.4	32.0
ZO-AdaMU (LoRA)	88.0	72.0	71.6	72.6	60.1	56.4	58.9	88.0	**83.2**	76.8	**32.4**
ZO-AdaMU (prefix)	88.0	61.8	**72.3**	**74.9**	56.5	58.2	61.9	86.0	82.8	**85.2**	30.4
Adam (FT) (12 ×)	92.0	70.8	83.9	77.1	63.5	70.1	71.1	79.0	74.1	84.9	31.3

Table 1: Experiments on OPT-13B (with 1,000 examples).

uncertainty can improve the stability of ZO-SGD. Therefore, this analysis highlights the positive gains on convergence rate from perturbation momentum and uncertainty.

Stable Convergence Rate

Classical descent lemma on SGD optimization highlights that the larger gradient covariance results in slower decrease in loss (Megerle et al. 2023).

Lemma 1 (Descent Lemma). *Let $\mathcal{L}(\boldsymbol{\theta})$ be ℓ-smooth (Wang and Xu 2019). For any unbiased gradient estimate $\boldsymbol{g}(\boldsymbol{\theta}, \mathcal{B})$,*

$$\begin{aligned}\mathbb{E}\left[\mathcal{L}\left(\boldsymbol{\theta}_{t-1}\right)|\boldsymbol{\theta}_t\right] - \mathcal{L}\left(\boldsymbol{\theta}_t\right) \leq & -\eta\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}_t\right)\right\|^2 \\ & + \frac{1}{2}\eta^2\ell\cdot\mathbb{E}\left[\left\|\boldsymbol{g}\left(\boldsymbol{\theta}, \mathcal{B}_t\right)\right\|^2\right]\end{aligned} \tag{8}$$

The gradient norm plays important role in the descent lemma. We derive the gradient norms for MeZO and ZO-AdaMU respectively as below.

Lemma 2. *Let $\mathcal{B}$ be a random minibatch of size B, so the gradient norms of MeZO and ZO-AdaMU are*

$$\left\|\widehat{\nabla}\mathcal{L}\left(\boldsymbol{\theta}, \mathcal{B}\right)\right\|^2 \sim \mathcal{N}\left(\frac{d+n-1}{n}\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}, \mathcal{B}\right)\right\|, 1\cdot\epsilon^2\right) \tag{9}$$

where ϵ represents the perturbation sampling scale.

Thus,

$$\underbrace{\frac{\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}_t\right)\right\|^2}{\frac{d+n-1}{n}\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}_t\right)\right\|^2+\epsilon}}_{\eta^-_{MeZO}} \leqslant \frac{\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}_t\right)\right\|^2}{\left\|\boldsymbol{g}\left(\boldsymbol{\theta}_t, \mathcal{B}\right)\right\|^2} \leqslant \underbrace{\frac{\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}_t\right)\right\|^2}{\frac{d+n-1}{n}\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}_t\right)\right\|^2+\epsilon}}_{\eta^+_{MeZO}} \tag{10}$$

In ZO-AdaMU, the simulated perturbation includes two Gaussian distributions with variances α and $1-\alpha$ and smoothing parameter β_1.

$$\begin{aligned}\left\|\widehat{\nabla}\mathcal{L}\left(\boldsymbol{\theta}, \mathcal{B}\right)\right\|^2 \sim \mathcal{N}(&\frac{d+n-1}{n}\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}, \mathcal{B}\right)\right\|, \\ &\left(\beta_1^2\alpha^2+(1-\beta_1)^2(1-\alpha)^2\right)\epsilon^2)\end{aligned} \tag{11}$$

So that,

$$\begin{aligned}\eta^-_{AdaMU} &\leqslant \frac{\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}_t\right)\right\|^2}{\left\|\boldsymbol{g}\left(\boldsymbol{\theta}_t, \mathcal{B}\right)\right\|^2} \leqslant \eta^+_{AdaMU} \\ \eta^-_{AdaMU} &= \frac{\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}_t\right)\right\|^2}{\frac{d+n-1}{n}\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}_t\right)\right\|^2+\epsilon\sqrt{\beta_1^2\alpha^2+(1-\beta_1)^2(1-\alpha)^2}} \\ \eta^+_{AdaMU} &= \frac{\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}_t\right)\right\|^2}{\frac{d+n-1}{n}\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}_t\right)\right\|^2-\epsilon\sqrt{\beta_1^2\alpha^2+(1-\beta_1)^2(1-\alpha)^2}}\end{aligned} \tag{12}$$

As $\sqrt{\beta_1^2\alpha^2+(1-\beta_1)^2(1-\alpha)^2} < 1$, we can conclude that $\eta^-_{MeZO} < \eta^-_{AdaMU} \leqslant \eta^+_{AdaMU} < \eta^+_{MeZO}$ and $\eta_{MeZO} = \frac{n}{d+n-1}\eta_{SGD} < \eta_{AdaMU}$. Therefore, ZO-AdaMU has a faster convergence rate than MeZO optimization.

The uncertainty in simulated perturbation also decreases the local effective rank of the Hessian of the loss (Papyan 2018, 2020; Ghorbani, Krishnan, and Xiao 2019; Yao et al. 2020; Sagun et al. 2017; Wu et al. 2020).

Lemma 3. *Let $G\left(\boldsymbol{\theta}_t\right) = \max_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{B}}\left\|\nabla\mathcal{L}\left(\boldsymbol{\theta}_t; \{(\boldsymbol{x}, \boldsymbol{y})\}\right)\right\|$, for all $\boldsymbol{\theta}_t$ such that $\left\|\boldsymbol{\theta}-\boldsymbol{\theta}_t\right\| \leq \eta d G\left[\left(\boldsymbol{\theta}\right)\right]$ there is $\nabla^2\mathcal{L}\left(\boldsymbol{\theta}\right) \preceq \boldsymbol{H}\left(\boldsymbol{\theta}_t\right)$, therefore the maximum of effective rank of gradient is $tr\left(\boldsymbol{H}\left(\boldsymbol{\theta}_t\right)\right)/\left\|\boldsymbol{H}\left(\boldsymbol{\theta}_t\right)\right\|_{op} \approx r$.*

With the same parameter size d and minibatch size B, the averaged $\widehat{G}(\boldsymbol{\theta}_t)$ on gradient estimates of MeZO and ZO-AdaMU have

$$\widehat{G}_{MeZO}(\boldsymbol{\theta}_t) > \widehat{G}_{AdaMU}(\boldsymbol{\theta}_t) \tag{13}$$

and thus,

$$\frac{tr\left(\boldsymbol{H}_{MeZO}\left(\boldsymbol{\theta}_t\right)\right)}{\left\|\boldsymbol{H}_{MeZO}\left(\boldsymbol{\theta}_t\right)\right\|_{op}} \leqslant \frac{tr\left(\boldsymbol{H}_{AdaMU}\left(\boldsymbol{\theta}_t\right)\right)}{\left\|\boldsymbol{H}_{AdaMU}\left(\boldsymbol{\theta}_t\right)\right\|_{op}} \tag{14}$$

The above analysis proves that ZO-AdaMU has a faster speed than MeZO to decrease the loss at each step.

Better Global Convergence

The upper bound on expected average regret reflects whether the optimization method can converge to a local optimum (Shamir 2017; Zhuang et al. 2020).

Lemma 4. *The convergence of SGD optimization is commonly measured by the expected average regret,*

$$\mathbb{E}\left[R\left(T\right)\right] = \sum_{t=1}^{T}\left[f_t\left(\boldsymbol{\theta}_t\right) - f_t\left(\boldsymbol{\theta}^*\right)\right] \tag{15}$$

Task	SST-2	SST-5	SNLI	MNLI	RTE	TREC
Type	sentiment		natural language inference			topic
Zero-shot	79.0	35.5	50.2	48.8	51.4	32.0
Adam	91.9 (±1.8)	47.5 (±1.9)	77.5 (±2.6)	70.0 (±2.3)	66.4 (±7.2)	85.0 (±2.5)
Adam (LoRA)	91.4 (±1.7)	46.7 (±1.1)	74.9 (±4.3)	67.7 (±1.4)	66.1 (±3.5)	82.7 (±4.1)
Adam (prefix)	91.9 (±1.0)	47.7 (±1.1)	77.2 (±1.3)	66.5 (±2.5)	66.6 (±2.0)	85.7 (±1.3)
LP	91.3 (±0.5)	51.7 (±0.5)	80.9 (±1.0)	71.5 (±1.1)	73.1 (±1.5)	89.4 (±0.5)
MeZO	93.3 (±0.7)	53.2 (±1.4)	83.0 (±1.0)	78.3 (±0.5)	78.6 (±2.0)	94.3 (±1.3)
MeZO (LoRA)	93.4 (±0.4)	52.4 (±0.8)	84.0 (±0.8)	77.9 (±0.6)	77.6 (±1.3)	95.0 (±0.7)
MeZO (prefix)	93.3 (±0.1)	53.6 (±0.5)	84.8 (±1.1)	**79.8** (±1.2)	77.2 (±0.8)	94.4 (±0.7)
MeZO-Adam	93.3 (±0.6)	**53.9** (±0.8)	85.3 (±0.8)	**79.6** (±0.4)	79.2 (±1.2)	95.1 (±0.3)
ZO-AdaMU	**93.8** (±0.3)	53.7 (±0.8)	83.3 (±0.7)	78.5 (±1.3)	77.9 (± 2.0)	93.7 (±2.5)
ZO-AdaMU (LoRA)	93.1 (±0.6)	51.6 (±1.2)	84.4 (±0.5)	78.6 (±0.8)	78.2 (±1.2)	**95.2** (±0.5)
ZO-AdaMU (prefix)	93.4 (±0.4)	53.6 (±0.7)	**85.5** (±0.2)	79.4 (±1.0)	78.9 (±1.5)	95.0 (±0.7)

Table 2: Experiments on RoBERTa-large (350M parameters) that include zero-shot learning, linear probing (LP), full-parameter fune-tuning with Adam, MeZO and ZO-AdaMU, and parameter-efficient Fine-tuning (LoRA and prefix learning) with Adam, MeZO and ZO-AdaMU respectively. All reported numbers are averaged accuracy (standard deviation) over 5 times runs.

where $f_t(\boldsymbol{\theta}^*)$ is the best value with optimal solution $\boldsymbol{\theta}^*$ on t-th step.

Assume that the loss $\mathcal{L}(\boldsymbol{\theta})$ has bounded gradients that $\|\nabla\mathcal{L}_t(\boldsymbol{\theta})\|^2 \leq G$ and $\|\nabla\mathcal{L}_t(\boldsymbol{\theta})\|_\infty \leq G_\infty$, and the distance between any $\theta_t^{\texttt{MeZO}}$, $\theta_t^{\texttt{AdaMU}}$ generated by MeZO and ZO-AdaMU are both bounded as $\|\theta_n^{\texttt{MeZO}} - \theta_m^{\texttt{MeZO}}\|_2 \leq D$& $\|\theta_n^{\texttt{MeZO}} - \theta_m^{\texttt{MeZO}}\|_\infty \leq D_\infty$ and $\left\|\theta_n^{\texttt{AdaMU}} - \theta_m^{\texttt{AdaMU}}\right\|_2 \leq D$& $\left\|\theta_n^{\texttt{AdaMU}} - \theta_\infty^{\texttt{AdaMU}}\right\|_2 \leq D_\infty$ respectively for any $m, n \in \{1, \cdots, T\}$. The maximum smoothing parameters $\beta_1 = 0.9$ and $\beta_2 = 0.01$ in ZO-AdaMU and $\beta_1 = 0$, $\beta_2 = 0$ in MeZO respectively. ZO-AdaMU and MeZO achieve the following guarantees respectively, for all $T \geqslant 1$.

$$
\begin{aligned}
R_{\texttt{MeZO}}(T) &\leqslant \frac{D^2}{2\alpha}\sum_{i=1}^{d}\sqrt{T} + \alpha G_\infty \sum_{i=1}^{d}\|g_{1:T,i}\|_2 + \sum_{i=1}^{d}\frac{D_\infty^2 G_\infty}{2\alpha\lambda^2} \\
R_{\texttt{AdaMU}}(T) &\leqslant \frac{D^2}{0.2\alpha}\sum_{i=1}^{d}\sqrt{Tv_{T,i}} + \frac{1.9\alpha G_\infty}{0.099\times 7.1^2}\sum_{i=1}^{d}\|g_{1:T,i}\|_2 \\
&\quad + \sum_{i=1}^{d}\frac{D_\infty^2 G_\infty\sqrt{1-\beta_2}}{1.8\alpha(1-\lambda)^2} \\
R_{\texttt{MeZO}}(T) &> R_{\texttt{AdaMU}}(T)
\end{aligned}
\tag{16}
$$

Above regret bound analysis shows that ZO-AdaMU has a smaller expected average regret than the on of MeZO, which proves that ZO-AdaMU has a better global convergence than MeZO. This is also the reason why ZO-AdaMU achieves better generalization on LMs.

Experiment

Preliminary researches (Brown et al. 2020; Gao, Fisch, and Chen 2021; Schick and Schütze 2021) experimentally demonstrated that zeroth-order optimization only works with prompt learning on LLMs fine-tuning. All experiments in this section use prompts to train the LLMs with just forward passes fine-tuning (MeZO (Malladi et al. 2023a) and ZO-AdaMU) and back-propagation fine-tuning (Adam).

To evaluate the effectiveness of the proposed adaptive perturbation with momentum in ZO-AdaMU for LLMs fine-tuning, we conduct the same experiments as MeZO on both masked language model (MLM) pre-trained LLMs (like RoBERTa-large 350M) (Liu et al. 2019) and auto-regressive pre-trained LLMs (OPT-13B) (Zhang et al. 2022) in few-shot and many-shot settings with prompts. In addition, all optimization methods are explored on full-parameter, LoRA and prefix fine-tuning (Li and Liang 2021). Finally, we give the visualizations of ZO-AdaMU, MeZO and Adam on 6 popular test functions for optimization.

Please refer to our code [1] for the details about datasets and prompt templates in Tabels 1, 2 and 4, and hyper-parameters, grid searching for best values in Eq. 7 and experimental results to evaluate stable convergence.

Auto-Regressive Language Models

As the auto-regressive LLMs have become the predominant base models in NLP, like GPT-3.5, GPT-4 (Lin et al. 2023), LLaMA (Touvron et al. 2023) and ChatGLM (Du et al. 2022), we conduct experiments with OPT-13B on three NLP task paradigms - sentence classification, multiple choice and text generation. All benchmarks are selected from SuperGLUE (Wang et al. 2019) (includes COPA, SST-2, RTE, CB, WSC, WIC, MultiRC, ReCoRD), BoolQ (Clark et al. 2019), SQuAD (Rajpurkar et al. 2016) and DROP (Dua et al. 2019). The few-shot training, validation and test sets are randomly sampled from each dataset with numbers of $1,000$, 500 and $1,000$ respectively. The main results are listed in Table 1, and which can reach the following observations and summaries.

I. ZO-AdaMU has obvious advantages in complex reasoning tasks. Table 1 shows that ZO-AdaMU and its LoRA, prefix variants outperform the MeZO and Adam fine-tuned OPT on all multiple choice and text generation tasks. Specif-

[1] https://github.com/MathIsAll/ZO-AdaMU.git

ically, ZO-AdaMU and its LoRA, prefix variants outperform MeZO's results with 1.0%, 1.0% and 2.0% on COPA and 1.3%, 1.8% and 1.6% on ReCoRD respectively. Moreover, the best F1 scores of ZO-AdaMU on SQuAD and DROP are both 1.0 higher than the best ones of MeZO. The advantage of ZO-AdaMU for Adam is more evident with gaps of 10.0, 9.1, 0.3 and 1.1 respectively.

II. ZO-AdaMU performs closest to back-propagation optimization methods across classification tasks. Experimental results in Table 1 show that the back-propagation optimization methods have more advantages than gradient-free methods on text classification tasks. Specifically, ZO-AdaMU obtains 3 best results out of 7 classification benchmarks, and beats MeZO counterparts on all tasks.

Tasks	AdaMU		Adam		AdamW		AdaMax		AadMM	
	δ	g	δ	g	δ	g	δ	g	δ	g
SST2	**92.1**	90.6	90.4	90.0	88.2	91.3	87.9	64.8	90.6	78.3
ReCoRD	**83.0**	81.2	73.1	71.3	83.0	79.1	77.6	82.0	80.3	80.4
SQuAD	**82.4**	82.1	82.0	81.0	77.3	68.4	80.0	68.3	79.7	73.8

Table 3: Ablation study by adapting different momentum schedules on perturbation (δ) and gradient (g) respectively.

We design an ablation study (Table 3) by adapting momentum schedules of Adam, AdamW, AdaMax, Rmsgrad on perturbation and gradients in ZO for LLMs prompt-tuning, respectively. These results show that our momentum schedule achieves the best results. In addition, the momentum schedules on perturbation are generally better than the ones on gradients, which verifies our idea that adapting momentum on the perturbation is the right way.

Masked Language Models

Experimental results on OPT-13B demonstrated the promising results of ZO-AdaMU on auto-regressive pre-rained LLMs. The second experiment extends ZO-AdaMU to the RoBERTa-large, a popular medium-size LM in the MLM family. This experiment follows the few-shot and many-shot settings from Gao, Fisch, and Chen (2021) and Malladi et al. (2023b), where $k = 512$ examples are sampled from per class for many-shot fine-tuning. The results are summarized in Table 2.

These results evaluate that (i) both ZO-AdaMU and MeZO significantly outperform the zero-shot and linear probing methods, which proves gradient-free ZOs really tune the LLMs. (ii) ZO-AdaMU and MeZO outperform Adam on 6 benchmarks, which demonstrates that the ZO-SGD methods effectively alleviate the over-fitting problem when the LLMs are fine-tuned on limited training data. (iii) The SPSA estimated gradient in Adam shows just a 0.43 average improvement, while ZO-AdaMU exhibits more dramatic increases with an average of 1.25.

The above experiments on MLM pre-trained LMs demonstrate that the concepts of adapting perturbation with momentum and uncertainty in SPSA are more suitable for ZO-SGD methods for LLMs fine-tuning.

Model	RoBERTa-large (350M)				OPT-13B
Task	**SST-2**	**SST-5**	**SNLI**	**TREC**	**SQuAD**
Zero-shot	79.00	35.50	50.20	32.00	46.20
MeZO	92.70	48.90	82.70	68.60	78.50
ZO-AdaMU	93.40	51.60	82.40	77.50	81.30

Table 4: ZO-AdaMU and MeZO with non-differentiable objectives. For classification tasks of SST-2, SST-5, SNLI and TREC, RoBERTa-large is optimized with full-parameter and accuracy on 500 examples; for SQuAD, OPT-13B is optimized with prefix and F1 on 1,000 examples.

Non-differentiable objectives

As our proposed ZO-AdaMU is also a gradient-free optimization method, we also conduct an experiment to evaluate ZO-AdaMU on RoBERTa-large and OPT with accuracy or F1 as objectives. Table 4 lists all results and they demonstrate that ZO-AdaMU outperforms its MeZO counterpart on both differentiable and non-differentiable objectives.

Memory usage

Hardware	Largest OPT that can fit		
	Adam	MeZO	ZO-AdaMU
1×A100 (80GB)	2.7B	30B	30B
2×A100 (160GB)	6.7B	66B	66B
4×A100 (320GB)	13B	66B	66B
8×A100 (640GB)	30B	175B	175B
1×V100 (30GB)	2.7B	6.7B	6.7B
2×V100 (60GB)	2.7B	13B	13B
4×V100 (120GB)	6.7B	30B	30B
8×V100 (240GB)	13B	66B	66B

Table 5: Largest OPT models that the mainstream hardwares of Nvidia A100 and V100 can tune.

As the storage of the expected moving average (EMA) for perturbation momentum, ZO-AdaMU slightly increases memory usage compared to MeZO. In Figure 3, we summarize the memory usage of zero-shot, in-context learning (ICL), prefix learning and full-parameter fine-tuning with Adam, MeZO and ZO-AdaMU. These statistics report the peak GPU memory usage by testing OPT models with Nvidia A100 GPUs on the SQuAD task (maximum length 2048 and minibatch size 1).

As shown in Figure 3 that MeZO exhibits the same memory consumption as zero-shot, which saves of up to 7 times of memory at least compared to back-propagation fine-tuning and 6 times compared to prefix fine-tuning. Even though our proposed ZO-AdaMU has a slight of memory usage increase compared to MeZO, it does not raise the requirements for mainstream hardware (like Nvidia A100 and V100) as shown in Table 5. This advantage enables training larger models within a fixed hardware budget, as illustrated in Figure 3. Specifically, using a single A100 GPU, ZO-AdaMU allows for tuning a model that is 11 times larger than what is feasible with full-parameter fine-tuning.

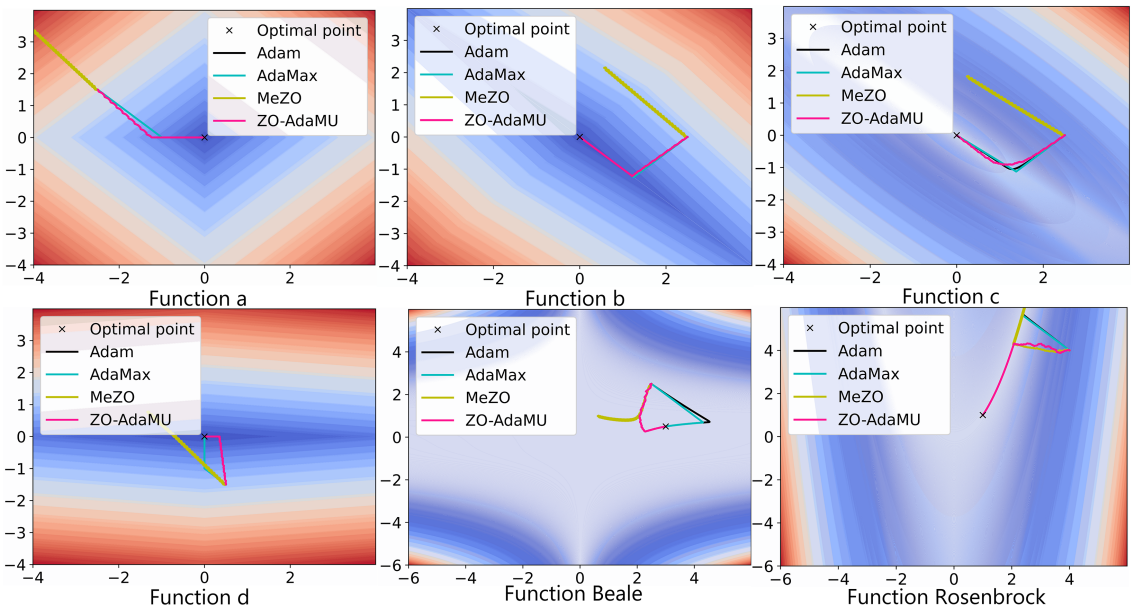

Figure 2: 2D trajectories of Adam, AdaMax, MeZO and ZO-AdaMU on 6 test functions. In all cases, ZO-AdaMU performs comparably to that first-order optimization methods, like Adam and AdaMax, but MeZO does not reach optimal points. We refer readers to video examples https://github.com/MathIsAll/ZO-AdaMU.git.

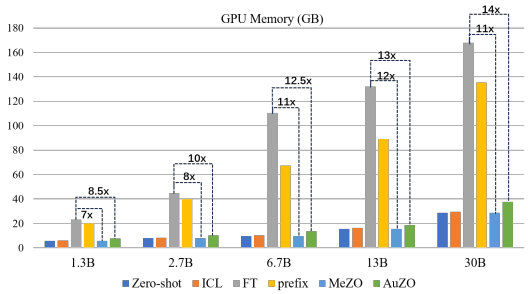

Figure 3: GPU memory consumption with OPT models fine-tuning on 2048 maximum length per example.

Trajectory Visualization on Test Functions

In this section, we validate the training trajectories of Adam, AdaMax, MeZO and ZO-AdaMU on 6 test functions, and the 2D trajectories are shown in Figure 2. These test functions are useful to evaluate characteristics of optimization algorithms, such as convergence rate, precision, robustness and general performance.

- a: $f(x,y) = |x| + |y|$ with global minimum $f(0,0) = 0$ and search domain $-3 \leqslant x, y \leqslant 3$;
- b: $f(x,y) = |x+y| + |x-y|/10$ with global minimum $f(0,0) = 0$ and search domain $-3 \leqslant x, y \leqslant 3$;
- c: $f(x,y) = (x+y)^2 + (x-y)^2/10$ with global minimum $f(0,0) = 0$ and search domain $-3 \leqslant x, y \leqslant 3$;
- d: $f(x,y) = |x|/10 + |y|$ with global minimum $f(0,0) = 0$ and search domain $-3 \leqslant x, y \leqslant 3$;
- Beale: $f(x,y) = (1.5 - x + xy)^2 + (2.25 - x + xy^2)^2 + (2.625 - x + xy^3)^2$ with global minimum $f(3, 0.5) = 0$ and search domain $-4.5 \leqslant x, y \leqslant 4.5$;
- Rosenbrock: $f(x,y) = 100(x-y^2)^2 + (1-y)^2$ with global minimum $f(1,1) = 0$ and search domain $-4.5 \leqslant x, y \leqslant 4.5$.

In all test functions, adapting perturbation with momentum in ZO-SGD reaches optimal points on all test functions, while MeZO optimizer fails to find any global minimum. Compared with momentum-based back-propagation optimizers, like Adam and AdaMax, ZO-AdaMU shows similar trajectories and reaches the optimal points on test functions of (a), (b) and (c). In addition, on test functions of (d), Beale and Rosenbrock, even though ZO-AdaMU shows different optimization trajectories, it reaches the optimal points with a faster speed than Adam and AdaMax.

Conclusion

We propose a ZO-AdaMU optimizer in this work, which adapts the momentum and uncertainty on simulated perturbation in the zeroth-order optimizer (ZO). To our knowledge, ZO-AdaMU is the first ZO-SGD optimizer that adapts the momentum on the stochastic approximation for simulated perturbation. Even though storing perturbation momentum requires a little extra memory cost compared with MeZO, ZO-AdaMU is consistent with MeZO for requirements of mainstream GPU hardware. We experimentally validate that ZO-AdaMU outperforms MeZO and back-propagation optimizers on convergence rate and generalization across various NLP tasks. Our visualizations prove that ZO-AdaMU performs comparably with Adam and AdaMax on popular test functions in machine learning.

Acknowledgments

This study is supported by grants from the National Key R&D Program of the Ministry of Science and Technology (Grant No. 2022ZD0116002), China Postdoctoral Science Foundation (No. 2023M741843), Shenzhen Science and Technology Plan No. KCXFZ20230731093001002), the Science and Technology Department of Guizhou Province (Grant No. Qiankehe Support[2022]General019), the National Key R&D Program of the Ministry of Science and Technology (Grant No. 2021ZD0113400), the National Social Science Foundation - Major Project (Grant No. 20ZD226), the Shenzhen Development and Reform Commission (Grant No. XMHT20190108009), the National Natural Science Foundation (Grant No. 62276075, 62106115, 62006062 and 62176076), the Guangdong Provincial Key Laboratory (Grant No. 2022B1212010005), the National Key RD Program of China (Grant No. 2022ZD0115305, 2021ZD0112905), the Major Key Project of PCL (Grant No. PCL2022D01), the Major Key Project of PCL (Grand No. PCL2021A06, PCL2022D01), the Key Laboratory of Intelligent Computing in Network Environment.

References

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Chen, X.; Liu, S.; Xu, K.; Li, X.; Lin, X.; Hong, M.; and Cox, D. 2019. Zo-adamm: Zeroth-order adaptive momentum method for black-box optimization. *Advances in neural information processing systems*, 32.

Clark, C.; Lee, K.; Chang, M.-W.; Kwiatkowski, T.; Collins, M.; and Toutanova, K. 2019. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. In *Proceedings of NAACL-HLT*, 2924–2936.

Ding, N.; Qin, Y.; Yang, G.; Wei, F.; Yang, Z.; Su, Y.; Hu, S.; Chen, Y.; Chan, C.-M.; Chen, W.; Yi, J.; Zhao, W.; Wang, X.; Liu, Z.; Zheng, H.-T.; Chen, J.; Liu, Y.; Tang, J.; Li, J.; and Sun, M. 2022. Delta Tuning: A Comprehensive Study of Parameter Efficient Methods for Pre-trained Language Models.

Du, Z.; Qian, Y.; Liu, X.; Ding, M.; Qiu, J.; Yang, Z.; and Tang, J. 2022. GLM: General Language Model Pretraining with Autoregressive Blank Infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 320–335.

Dua, D.; Wang, Y.; Dasigi, P.; Stanovsky, G.; Singh, S.; and Gardner, M. 2019. DROP: A Reading Comprehension Benchmark Requiring Discrete Reasoning Over Paragraphs. In *Proceedings of NAACL-HLT*, 2368–2378.

Fu, Z.; Yang, H.; So, A. M.-C.; Lam, W.; Bing, L.; and Collier, N. 2023. On the effectiveness of parameter-efficient fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 12799–12807.

Gao, T.; Fisch, A.; and Chen, D. 2021. Making pre-trained language models better few-shot learners. In *Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL-IJCNLP 2021*, 3816–3830. Association for Computational Linguistics (ACL).

Ghadimi, S.; and Lan, G. 2013. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4): 2341–2368.

Ghorbani, B.; Krishnan, S.; and Xiao, Y. 2019. An investigation into neural net optimization via hessian eigenvalue density. In *International Conference on Machine Learning*, 2232–2241. PMLR.

Hu, E. J.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W.; et al. 2021. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

Kingma, D.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*. San Diega, CA, USA.

Li, X. L.; and Liang, P. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 4582–4597.

Lin, J. C.; Younessi, D. N.; Kurapati, S. S.; Tang, O. Y.; and Scott, I. U. 2023. Comparison of GPT-3.5, GPT-4, and human user performance on a practice ophthalmology written examination. *Eye*, 1–2.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Lv, K.; Yang, Y.; Liu, T.; Gao, Q.; Guo, Q.; and Qiu, X. 2023. Full Parameter Fine-tuning for Large Language Models with Limited Resources. *arXiv preprint arXiv:2306.09782*.

Malladi, S.; Gao, T.; Nichani, E.; Damian, A.; Lee, J. D.; Chen, D.; and Arora, S. 2023a. Fine-Tuning Language Models with Just Forward Passes. *arXiv preprint arXiv:2305.17333*.

Malladi, S.; Wettig, A.; Yu, D.; Chen, D.; and Arora, S. 2023b. A kernel-based view of language model fine-tuning. In *International Conference on Machine Learning*, 23610–23641. PMLR.

Maryak, J. L.; and Chin, D. C. 2001. Global random optimization by simultaneous perturbation stochastic approximation. In *Proceedings of the 2001 American control conference.(Cat. No. 01CH37148)*, volume 2, 756–762. IEEE.

Megerle, D.; Otto, F.; Volpp, M.; and Neumann, G. 2023. Stable Optimization of Gaussian Likelihoods.

Papyan, V. 2018. The full spectrum of deepnet hessians at scale: Dynamics with sgd training and sample size. *arXiv preprint arXiv:1811.07062*.

Papyan, V. 2020. Traces of class/cross-class structure pervade deep learning spectra. *The Journal of Machine Learning Research*, 21(1): 10197–10260.

Rajbhandari, S.; Rasley, J.; Ruwase, O.; and He, Y. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–16. IEEE.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2383–2392.

Sagun, L.; Evci, U.; Güney, V. U.; Dauphin, Y.; and Bottou, L. 2017. Empirical Analysis of the Hessian of Over-Parametrized Neural Networks. *ArXiv*, abs/1706.04454.

Schick, T.; and Schütze, H. 2021. Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 255–269.

Shamir, O. 2017. An optimal algorithm for bandit and zero-order convex optimization with two-point feedback. *The Journal of Machine Learning Research*, 18(1): 1703–1713.

Shamir, O.; and Zhang, T. 2013. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *International conference on machine learning*, 71–79. PMLR.

Sun, S.; Liu, Y.; Iter, D.; Zhu, C.; and Iyyer, M. 2023a. How does in-context learning help prompt tuning? *arXiv preprint arXiv:2302.11521*.

Sun, X.; Ji, Y.; Ma, B.; and Li, X. 2023b. A Comparative Study between Full-Parameter and LoRA-based Fine-Tuning on Chinese Instruction Data for Instruction Following Large Language Model. *arXiv preprint arXiv:2304.08109*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Wang, A.; Pruksachatkun, Y.; Nangia, N.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Wang, D.; and Xu, J. 2019. Differentially private empirical risk minimization with smooth non-convex loss functions: A non-stationary view. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 1182–1189.

Wei, J.; Tay, Y.; Bommasani, R.; Raffel, C.; Zoph, B.; Borgeaud, S.; Yogatama, D.; Bosma, M.; Zhou, D.; Metzler, D.; Chi, E. H.; Hashimoto, T.; Vinyals, O.; Liang, P.; Dean, J.; and Fedus, W. 2022. Emergent Abilities of Large Language Models. *Transactions on Machine Learning Research*. Survey Certification.

Wu, Y.; Zhu, X.; Wu, C.; Wang, A.; and Ge, R. 2020. Dissecting hessian: Understanding common structure of hessian in neural networks. *arXiv preprint arXiv:2010.04261*.

Yao, Z.; Gholami, A.; Keutzer, K.; and Mahoney, M. W. 2020. Pyhessian: Neural networks through the lens of the hessian. In *2020 IEEE international conference on big data (Big data)*, 581–590. IEEE.

Zhang, S.; Roller, S.; Goyal, N.; Artetxe, M.; Chen, M.; Chen, S.; Dewan, C.; Diab, M.; Li, X.; Lin, X. V.; et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Zhuang, J.; Tang, T.; Ding, Y.; Tatikonda, S. C.; Dvornek, N.; Papademetris, X.; and Duncan, J. 2020. Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. *Advances in neural information processing systems*, 33: 18795–18806.

Appendix A

A.1 Prompt templates

The following Table 1 describes the prompt templates on tasks in our experiments, which are set same as (Malladi et al. 2023) and (Gao, Fisch, and Chen 2021).

Dataset	C	Type	Prompt	Labels
SST-2	2	SC	$< \mathcal{S}_1 >$ It was [MASK] .	{great, terrible}
SST-5	5	SC	$< \mathcal{S}_1 >$ It was [MASK] .	{great, good, okay, bad, terrible}
TREC	6	TC	[MASK] : $< \mathcal{S}_1 >$	{description, expression, entity, human, location, number}
MNLI	3	NLI	$< \mathcal{S}_1 >$? [MASK] , $< \mathcal{S}_2 >$	{Yes, Maybe, No}
SNLI	3	NLI	$< \mathcal{S}_1 >$? [MASK] , $< \mathcal{S}_2 >$	{Yes, Maybe, No}
RTE	2	NLI	$< \mathcal{S}_1 >$? [MASK] , $< \mathcal{S}_2 >$	{Yes, No}

Table 1: The prompts used in RoBERTa-large experiments (Table 2 and Figure 2). The prompts are adapted from (Malladi et al. 2023) and (Gao, Fisch, and Chen 2021), in which a template and a set of label words that can fill in the [MASK] token. $< \mathcal{S}_1 >$ and $< \mathcal{S}_2 >$ refer to the first and the second (if any) input sentence.

Table 2 demonstrates the prompts used for OPT in Table 1. These prompts are categorized into three types of tasks: classification (CLS), multiple-choice (MulC), and question answering (QA). Prompts are adopted from PromptSource (Bach et al. 2022) for GPT-3 (Brown et al. 2020) with minor changes.

Dataset	Type	Prompts
SST-2	CLS	<text> It was terrible/great
RTE	CLS	<premise> Does this mean that "<hypothesis>" is true? Yes or No? Yes/No
CB	CLS	Suppose <premise> Can we infer that "<hypothesis>"? Yes, No, or Maybe? Yes/No/Maybe
BoolQ	CLS	<passage> <question>? Yes/No
WSC	CLS	<text> In the previous sentence, does the pronoun "<span2>" refer to "<span1>"? Yes or No? Yes/No
WIC	CLS	Does the word "<word>" have the same meaning in these two sentences? Yes, No? <sent1>; <sent2> Yes/No
MultiRC	CLS	<paragraph> Question: <question> I found this answer "<answer>". Is that correct? Yes or No? Yes/No
COPA	MulC	<premise> so/because <candidate>
ReCoRD	MulC	<passage> <query>.replace(''@placeholder'', <candidate>)
SQuAD	QA	Title: <title> Context: <context> Question: <question> Answer:
DROP	QA	Passage: <context> Question: <question> Answer:

Table 2: The prompts used for OPT in Table 1, which are adopted from PromptSource (Bach et al. 2022) on GPT-3 (Brown et al. 2020) with minor changes. These prompts are categorized into three types of tasks: classification (CLS), multiple-choice (MulC), and question answering (QA). <text> represents input from the dataset and Yes represents label words. For inference on multiple choice tasks, we put in different candidates in the prompt and calculate the average log-likelihood for each candidate, and choose the candidate with the highest score. For inference on QA tasks, we use greedy decoding to generate the answer.

Appendix B

Hyperparameters

We use the hyperparameters in Table 3 for ZO-AdaMU, MeZO and Adam experiments on both OPT (Table 1) and RoBERTa-large (Table 2 and Figure 2).

Experiment	Hyperparameters	Best values	Searching grids
MeZO (FT)	Batch size	64	64
	Learning rate	$1e-6$	$\{1e-7, 1e-6, 1e-5\}$
	ϵ	$1e-3$	$1e-3$
	Weight Decay	0	0
MeZO (prefix)	Batch size	64	64
	Learning rate	$1e-4$	$\{1e-5, 1e-4, 1e-3\}$
	ϵ	$1e-2$	$\{1e-3, 1e-2, 1e-1\}$
	Weight Decay	0	0
	# prefix tokens	5	5
MeZO (LoRA)	Batch size	64	64
	Learning rate	$1e-2$	$\{1e-3, 1e-2, 1e-1\}$
	ϵ	$1e-1$	$\{1e-3, 1e-2, 1e-1\}$
	Weight Decay	0	0
	(γ, α)	(8, 16)	(8, 16)
Adam (FT)	Batch size	64	64
	Initialized learning rate	$1e-5$	$1e-5$
	Weight Decay	0	0
Adam (prefix)	Batch size	64	64
	Learning rate	$1e-5$	$1e-5$
	Weight Decay	0	0
	# prefix tokens	5	5
Adam (LoRA)	Batch size	64	64
	Learning rate	$1e-5$	$1e-5$
	Weight Decay	0	0
ZO-AdaMU (FT)	Batch size	64	64
	Learning rate	$1e-4$	$\{1e-6, 1e-4, 1e-2\}$
	ϵ	$1e-3$	$\{1e-3, 1e-2, 1e-1\}$
	(T^1, T^2)	$(1024, 16000)$	$(1024, 16000)$
	$(\varphi^\alpha, \varphi^{\beta_1}, \varphi^{\beta_2})$	$(1.0, 0.1, 1.5)$	$(1.0, 0.1, 1.5)$
	Weight Decay	0	0
ZO-AdaMU (prefix)	Batch size	64	64
	Learning rate	$1e-4$	$\{1e-6, 1e-4, 1e-2\}$
	ϵ	$1e-2$	$\{1e-3, 1e-2, 1e-1\}$
	(T^1, T^2)	$(1024, 16000)$	$(1024, 16000)$
	$(\varphi^\alpha, \varphi^{\beta_1}, \varphi^{\beta_2})$	$(1.0, 0.1, 1.5)$	$(1.0, 0.1, 1.5)$
	Weight Decay	0	0
	# prefix tokens	5	5
ZO-AdaMU (LoRA)	Batch size	64	64
	Learning rate	$1e-2$	$\{1e-6, 1e-4, 1e-2\}$
	ϵ	$1e-1$	$\{1e-3, 1e-2, 1e-1\}$
	(T^1, T^2)	$(1024, 16000)$	$(1024, 16000)$
	$(\varphi^\alpha, \varphi^{\beta_1}, \varphi^{\beta_2})$	$(1.0, 0.1, 1.5)$	$(1.0, 0.1, 1.5)$
	Weight Decay	0	0

Table 3: The best hyperparameters used for both OPT and RoBERTa-large experiments, both ZO-AdaMU and MeZO use a constant learning rate schedule, and Adam uses linear scheduling. All Adam experiments use 1K steps, ZO-AdaMU and MeZO experiments use 100K steps. We check validation performance every 1/10 total training steps.

Appendix C

3D Gif of trajectory

Please refer to the attachment `3d_objective_a.gif`, `3d_objective_b.gif`, `3d_objective_c.gif`, `3d_objective_d.gif`, `3d_beale_function.gif`, `3d_rosenbrock.gif` for the trajectories for test functions in Figure 3.

Appendix D

Better local optimal

The experimental results showed in Figure 1 are conducted on OPT-1.3B with ZO-AdaMU and MeZo respectively. The training and validation samples are taken same as Table 1 and 2. Results demonstrate that ZO-AdaMU could optimize the large language models better in few-shot learning setting. The reason why the validation loss rises after $3,000$ and $2,700$ steps on SQuAD and BoolQ is that MeZO optimized OPT over-fits on training data, especially in the few-shot setting. This also proved that the adapting perturbation with momentum and uncertainty effectively improves the convergence, and improves the LLMs' generalization for few-shot fine-tuning.

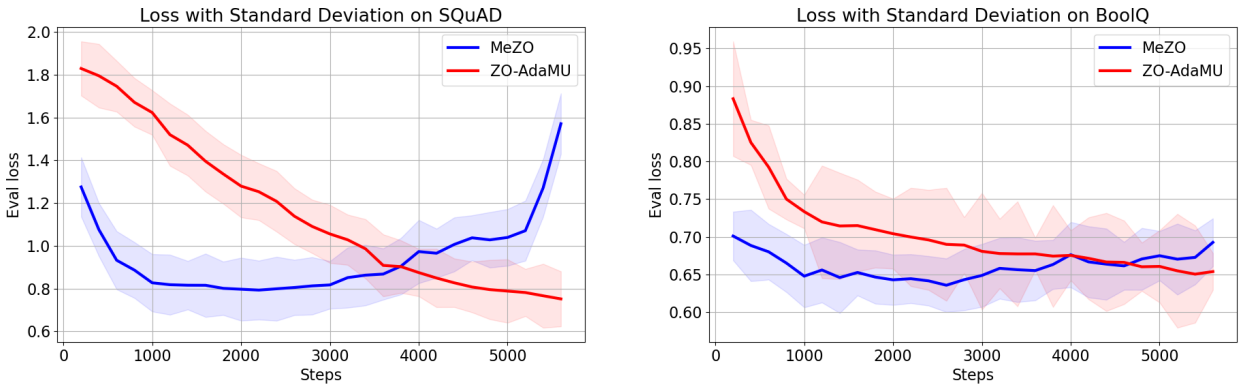

Figure 1: ZO-AdaMU converges on better local optimal point compared to MeZO. This experiment is conducted on OPT-1.3B language model on SQuAD and BoolQ tasks with $1,000$ examples randomly sampled for training, and the listed losses are evaluated from 500 examples randomly sampled from validation data.

References

Bach, S. H.; Sanh, V.; Yong, Z.-X.; Webson, A.; Raffel, C.; Nayak, N. V.; Sharma, A.; Kim, T.; Bari, M. S.; Fevry, T.; Alyafeai, Z.; Dey, M.; Santilli, A.; Sun, Z.; Ben-David, S.; Xu, C.; Chhablani, G.; Wang, H.; Fries, J. A.; Al-shaibani, M. S.; Sharma, S.; Thakker, U.; Almubarak, K.; Tang, X.; Tang, X.; Jiang, M. T.-J.; and Rush, A. M. 2022. PromptSource: An Integrated Development Environment and Repository for Natural Language Prompts. arXiv:2202.01279.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Gao, T.; Fisch, A.; and Chen, D. 2021. Making pre-trained language models better few-shot learners. In *Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL-IJCNLP 2021*, 3816–3830. Association for Computational Linguistics (ACL).

Malladi, S.; Gao, T.; Nichani, E.; Damian, A.; Lee, J. D.; Chen, D.; and Arora, S. 2023. Fine-Tuning Language Models with Just Forward Passes. *arXiv preprint arXiv:2305.17333*.