PaZO: Preconditioned Accelerated Zeroth-Order Optimization for Fine-Tuning LLMs

Anonymous Author(s)
Affiliation
Address
email

Abstract

This paper introduces PaZO, a preconditioned accelerated zeroth-order optimization algorithm for fine-tuning large language models (LLMs). First, we theoretically demonstrate the necessity of preconditioning in zeroth-order optimization, proving that zeroth-order stochastic gradient descent (ZO-SGD) alone fails to achieve the ideal convergence rate. Building on this, we propose a Preconditioned Simultaneous Perturbation Stochastic Approximation (PSPSA) and theoretical version of PaZO, and demonstrate that setting the order of preconditioner as $-1/2$ in PSPSA yields the improved convergence rate for PaZO. Moreover, we design a practical version of PaZO that stabilizes training via diagonal Hessian estimate and moving average technique. Extensive experiments on diverse downstream tasks with models like RoBERTa-large and OPT show PaZO's effectiveness. Compared to other zeroth-order baselines, PaZO achieves better performance across models and tasks.

1 Introduction

Fine-tuning pre-trained large language models (LLMs) has become one of the dominant methodologies for adapting models to specialized downstream tasks [19] and aligning them with human instructional preferences [42]. However, as models are scaled up [1], the memory overhead extremely increases during fine-tuning, since computing gradients during backpropagation needs to cache model activations and historical gradients (e.g., for Adam-based optimization [28]). Parameter-efficient fine-tuning (PEFT) methods [29, 31, 23] reduce memory overhead by fine-tuning only a small number of extra parameters but still need to cache large quantities of activations. Recently, zeroth-order optimization algorithms (ZO) [37, 59, 58] have enabled the fine-tuning of LLMs with billions of parameters on a single consumer-grade GPU, due to their requirement for only forward passes to estimate gradients, without backpropagation and the storage of activations. Lightweight memory has solidified its role as a critical methodology for fine-tuning tasks in resource-constrained scenarios.

As research on zeroth-order optimization methods for fine-tuning LLMs advances, whether preconditioning zeroth-order algorithms with higher-order information can enhance optimization efficiency has become a pivotal challenge, since adaptive first-order optimizers such as Adam [28] and AdamW [35], which can be regarded as preconditioned algorithms with $(\mathrm{diag}\{\mathbf{g} \circ \mathbf{g}\})^{-1/2}$ as a preconditioner, show improvement on convergence speed. However, for zeroth-order optimization, one cannot directly estimate the Hessian by first-order information. Direct adaptation of Adam to zeroth-order algorithms (e.g., ZO-Adam [58]) introduces large variances and has a significant impact on the fine-tuning performance [59]. Moreover, Hessian-informed perturbation for estimating zeroth-order information [59, 55] is a significant methodological advancement, but how to incorporate Hessian information into the perturbation process to obtain the best convergence speed and performance remains a significant challenge.

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

When we delve into and rethink the preconditioned zeroth-order optimization problems, the more pressing challenge lies in whether preconditioned zeroth-order optimization methods can truly achieve a provable convergence rate from a theoretical perspective. This problem may appear counterintuitive, but mature theoretical research [24, 17] on first-order methods has substantiated the following facts: for least squares regression, only SGD can achieve the near-optimal convergence rate $\tilde{\mathcal{O}}(d/T)$ and match the lower bound when ignoring the logarithmic term, which indicates that at least for this problem, preconditioning techniques provide no improvement on convergence, as SGD has already attained the information-theoretic limit of the problem. Therefore, whether this conclusion for zeroth-order optimization remains determines the effect of preconditioning techniques in zeroth-order optimization. Moreover, even if we posit that precondition holds effectiveness for zeroth-order optimization, how to appropriately apply preconditioning techniques emerges as another challenge. Specifically, determining the optimal order of the preconditioner to guarantee the fastest convergence rate becomes a critical consideration. Finally, from the practical perspective, how to estimate Hessian information through zeroth-order perturbation stochastic approximation to integrate abundant information, ensure stability and control memory overhead is also a challenge in practice. Based on the three above, we think that the following three problems demand reasonable resolution in preconditioned zeroth-order optimization for fine-tuning LLMs:

> *A*. Do we truly need preconditions in zeroth-order optimization?
>
> *B*. If the answer to question *A* is "yes", how to achieve the fastest convergence by selecting the optimal order of the preconditioner?
>
> *C*. How to effectively estimate Hessian information through zeroth-order perturbations in practice and improve fine-tuned model performance on downstream tasks?

In this paper, we provide reasonable answers to the three questions above. We propose a preconditioned accelerated zeroth-order optimization algorithm PaZO, with a theoretical guarantee to obtain a faster convergence rate by selecting the optimal order of preconditioner, and better empirical performance on a wide range of downstream tasks for fine-tuning LLMs. Our contributions are:

1. (Answer to Question *A*.) We construct a general Preconditioned Simultaneous Perturbation Stochastic Approximation (PSPSA) and corresponding algorithm PaZO (Theoretical Form 3.2) with any given order of Hessian information $\mathbf{H}^{-\alpha}$. Our theoretical analysis on quadratic functions in Theorem 3.5 demonstrates that only ZO-SGD ($\alpha = 0$) **cannot** achieve the fastest convergence rate. We need preconditions in zeroth-order optimization.
2. (Answer to Question *B*.) We provide the convergence analysis of PaZO for general objective functions. The result in Theorem 3.8 demonstrates that PaZO can achieve the fastest convergence rate if and only if we select $\alpha = 1/2$. In other words, we need to use $\mathbf{H}^{-1/2}$ in PSPSA (or $\mathbf{H}^{-1}$ as the preconditioner) to accelerate zeroth-order optimization.
3. (Answer to Question *C*.) We propose PaZO (Practical Form, Algorithm 1) for fine-tuning LLMs, with unbiased diagonal Hessian estimation incorporating current zeroth-order gradient information and moving average techniques to ensure stability in practice. We conduct extensive experiments across different models (RoBERTa-large, OPT-1.3B), different methods (FT, LoRA, prefix), and different downstream tasks to verify the effect of the PaZO. Results show PaZO achieves better performance across models, tasks and PEFT methods.

Notations. Let $\mathcal{O}(\cdot)$ and $\Omega(\cdot)$ denote upper and lower bounds, respectively, with a universal constant, while $\tilde{\mathcal{O}}(\cdot)$ and $\tilde{\Omega}(\cdot)$ ignore polylogarithmic dependencies. For functions f and g: $f \lesssim g$ denotes $f = \tilde{\mathcal{O}}(g)$; $f \gtrsim g$ denotes $f = \tilde{\Omega}(g)$; $f \asymp g$ indicates $g \lesssim f \lesssim g$. We use $\lambda_{max}(\cdot)$ and $\lambda_{min}(\cdot)$ to denote the largest and smallest eigenvalue of a matrix, respectively. Let $\|\boldsymbol{\theta}\|_{\mathbf{A}}$ denote the Mahalanobis (semi) norm where $\mathbf{A}$ is a positive semi-definite matrix as $\|\boldsymbol{\theta}\|_{\mathbf{A}} = \sqrt{\boldsymbol{\theta}^\top \mathbf{A} \boldsymbol{\theta}}$. We use $\boldsymbol{\theta}^*$ to denote the minimizer, i.e. $\boldsymbol{\theta}^* \triangleq \operatorname{argmin}_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$.

2 Related Work

Zeroth-order Optimization: Zeroth-order optimization, is to estimate the gradient by just forward passes. A substantial body of theoretical research has been devoted to the detailed analysis of convergence rates in zeroth-order optimization in convex settings [3, 16, 26, 39, 44, 46] and non-convex [53]. Representative method SPSA [48] demonstrates strong performance in challenging settings like non-convex multi-agent optimization [21, 50] and black-box adversarial example generation [11, 10, 33]. Notably, MeZO [37] pioneers the adaptation of classical ZO-SGD for LLM fine-tuning, matching conventional performance while drastically cutting memory consumption. Then various following works [58, 59, 12, 49] try to improve zeroth-order optimizers for efficient fine-tuning. However, whether and how precondition works in zeroth-order optimization is still lack of discussion.

Enhanced Optimizers with Hessian: Researchers focus on how to incorporate second-order information to provide acceleration for gradient descent during the training. For example, [9, 40] utilized curvature information as the preconditioner; [38] applied diagonal Hessian as the preconditioner; [36] estimated the Hessian information with conjugate gradient. Sophia [32] introduced a lightweight estimate of the diagonal Hessian for pre-training. However, these methods can only be used for first-order methods with a heavy GPU-memory overhead. HiZOO [59] has been proposed as a preconditioned zeroth-order optimizer for fine-tuning LLMs. However, how to effectively leverage preconditioning information in zeroth-order optimization to accelerate convergence remains understudied.

3 Theoretical Insights of PaZO

Preconditioned methods in first-order optimization have been generally studied [40, 4, 28, 32]. However, few works discuss the necessity, potential and limitation of preconditioned zeroth-order optimization. In this section, we try to clarify two questions below from the theoretical perspective.

> *A*. Do we truly need preconditions in zeroth-order optimization?
>
> *B*. If the answer to question *A* is "yes", how to achieve the fastest convergence by selecting the optimal order of the preconditioner?

We provide theoretical insights into the two questions *A* and *B*. First, we show the necessity of using preconditions in zero-order optimization, since only ZO-SGD [48] cannot achieve the potential ideal convergence rate $\tilde{\mathcal{O}}(d^2/T)$ for least squares (as stated in Theorem 3.5), while the first-order SGD can match the optimal rate $\tilde{\mathcal{O}}(d/T)$ without preconditions [17]. This difference indicates that preconditions play a key role in ZO, especially. Second, we propose a general Preconditioned Simultaneous Perturbation Stochastic Approximation (PSPSA) using $\mathbf{H}^{-\alpha}$ as preconditioner with any given order α and Hessian $\mathbf{H}$ to extend traditional SPSA [48] for zeroth-order gradient estimate. We provide the convergence analysis of the preconditioned zeroth-order optimization with PSPSA in Theorem 3.8. The results explicitly direct us to choose the optimal α to obtain the fastest rate.

3.1 Problem Setup

We consider the standard stochastic unconstrained minimization problem as:

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^d} f(\boldsymbol{\theta}) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[F(\boldsymbol{\theta};(\mathbf{x},y))], \tag{1}$$

where the expectation is taken over the data distribution $(\mathbf{x},y)\sim\mathcal{D}$. Given the Hessian matrix $\mathbf{H}_t$ at the decision point $\boldsymbol{\theta}_t$, we first define the following general Preconditioned Simultaneous Perturbation Stochastic Approximation (PSPSA) as:

Definition 3.1 (Preconditioned Simultaneous Perturbation Stochastic Approximation (PSPSA))**.** *Given a model with parameters $\boldsymbol{\theta}\in\mathbb{R}^d$ and the loss function F, PSPSA estimates the zeroth-order stochastic gradient $\tilde{\nabla}F(\boldsymbol{\theta}_t)$ at $(\mathbf{x}_t,y_t)$ as*

$$\tilde{\nabla}F(\boldsymbol{\theta}_t;(\mathbf{x}_t,y_t)) = \frac{F(\boldsymbol{\theta}_t+\mu\mathbf{H}_t^{-\alpha}\mathbf{u};(\mathbf{x}_t,y_t)) - F(\boldsymbol{\theta}_t-\mu\mathbf{H}_t^{-\alpha}\mathbf{u};(\mathbf{x}_t,y_t))}{2\mu}\cdot\mathbf{H}_t^{-\alpha}\mathbf{u}, \tag{2}$$

where $\mathbf{u}\in\mathbb{R}^d$ and $\mathbf{u}\sim\mathcal{N}(0,\mathbf{I}_d)$, μ is the perturbation scale, $\mathbf{H}_t$ is the Hessian matrix at $\boldsymbol{\theta}_t$, and $\alpha\in\left[-\frac{1}{2},\frac{1}{2}\right]$ is the precondition order.

With the estimated zeroth-order stochastic gradient generated by PSPSA, the preconditioned zeroth-order optimization algorithm can be stated as follows:

Definition 3.2 (Preconditioned Accelerated Zeroth-order Optimization, PaZO (Theoretical Form))**.** *PaZO is an optimizer with learning rate η that updates parameters as*

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta\tilde{\nabla}F(\boldsymbol{\theta}_t; (\mathbf{x}_t, y_t)), \tag{3}$$

where $\tilde{\nabla}F(\boldsymbol{\theta}_t; (\mathbf{x}_t, y_t))$ is the PSPSA gradient estimate at $\boldsymbol{\theta}_t$ with $\mathbf{H}_t$.

PSPSA and PaZO can be regarded as the general preconditioned extension of the existing zeroth-order perturbation approximation and algorithms. Intuitively, ignoring the higher-order infinitesimal term of μ, we obtain the expectation of the PSPSA gradient estimate as

$$\mathbb{E}\left[\tilde{\nabla}F(\boldsymbol{\theta}_t; (\mathbf{x}_t, y_t))\right] = \mathbb{E}_{\mathbf{u}}\left[\frac{2\mu\nabla f^\top(\boldsymbol{\theta}_t)\cdot\mathbf{H}_t^{-\alpha}\mathbf{u}}{2\mu}\cdot\mathbf{H}_t^{-\alpha}\mathbf{u}\right] = \mathbf{H}_t^{-2\alpha}\nabla f(\boldsymbol{\theta}_t), \tag{4}$$

which indicates that the PSPSA gradient estimate is equivalent to a $\mathbf{H}_t^{-2\alpha}$ preconditioned gradient. When $\alpha = 0$, PSPSA degenerates to SPSA [48] and PaZO is reduced to ZO-SGD. When $\alpha = -1/2$, PaZO is equivalent to the representative Hessian-informed zeroth-order method HiZOO [59].

We introduce the assumption below to construct the relation between the outer product of the gradient and the Hessian for our analysis.

Assumption 3.3 (Unbiased Estimate of Hessian)**.** *We assume that the expectation of the outer product of $F(\boldsymbol{\theta}^*, (\mathbf{x}, y))$ is the unbiased estimate of $\mathbf{H}^*$ as:*

$$\mathbb{E}\left[\nabla F(\boldsymbol{\theta}^*; (\mathbf{x}, y))\nabla^\top F(\boldsymbol{\theta}^*; (\mathbf{x}, y))\right] = \mathbf{H}^*, \tag{5}$$

where $\boldsymbol{\theta}^*$ is a minimizer of the objective $f(\boldsymbol{\theta})$, and $\mathbf{H}^*$ is the Hessian defined at $\boldsymbol{\theta}^*$.

Assumption 3.3 is a common assumption when considering stochastic gradient descent [17, 24, 5, 25], especially for least squares regression [17, 24], whose Hessian is fixed and can be exactly calculated.

3.2 Case Study: Least Squares Regression

First, we try to provide an intuitive answer to the question *A*. We consider a representative case of f: least squares regression, whose optimization dynamic can be clear and meticulously calculated due to the fixed Hessian as:

$$F(\boldsymbol{\theta}; (\mathbf{x}, y)) = \frac{1}{2C}\left(y - \langle\boldsymbol{\theta}, \mathbf{x}\rangle\right)^2. \tag{6}$$

We have access to stochastic gradients zeroth-order obtained by PSPSA with sampling a new example $(\mathbf{x}_t, y_t) \sim \mathcal{D}$. These examples satisfy

$$y = \langle\boldsymbol{\theta}^*, \mathbf{x}\rangle + \epsilon,$$

where ϵ is a noise on the example pair with $\mathbb{E}[\epsilon] = 0$ and $\mathbb{E}[\epsilon^2] = \sigma^2$, and $\boldsymbol{\theta}^*$ is a minimizer of the objective. Note that the Hessian of the objective $\mathbf{H}^* \stackrel{\text{def}}{=} \nabla^2 f(\boldsymbol{\theta}) = \frac{1}{C}\mathbb{E}[\mathbf{x}\mathbf{x}^\top]$. The following estimate holds

$$\mathbb{E}\left[\nabla F(\boldsymbol{\theta}^*; (\mathbf{x}, y))\nabla^\top F(\boldsymbol{\theta}^*; (\mathbf{x}, y))\right] = \frac{1}{C^2}\mathbb{E}[\epsilon^2\mathbf{x}\mathbf{x}^\top] = \frac{\sigma^2}{C}\mathbf{H}^*. \tag{7}$$

By setting $C = \sigma^2$, we exactly obtain the result in Assumption 3.3. The analytical tractability of (6) offers deeper theoretical insights. Specifically, previous studies [17] demonstrate that for first-order algorithms the *optimal* rate achieves $\tilde{\mathcal{O}}(d/T)$ and construct the lower bound, where d is the dimension of problems and T is the iteration steps. Moreover, the studies show that *only SGD* can match the near-optimal rate with only the difference of logarithmic terms. In other words, for least squares regression and first-order stochastic algorithms, only SGD is enough with any precondition making no effect of acceleration. When turning to zeroth-order optimization, intuitively, we think the *ideal convergence rate* achieves $\tilde{\mathcal{O}}(d^2/T)$ since in zeroth-order optimization we can only access one-dimension information per step. Varieties of theoretical studies of zeroth-order algorithms [2, 41] also show d times slower convergence rate than first-order ones. However, the results stated in Theorem 3.5 indicate that only ZO-SGD is not enough.

Assumption 3.4 (Fourth Moment Conditions)**.** *Suppose $\mathbf{B}$ is a positive semi-definite matrix, and consider data vector $\mathbf{x}$. It satisfies $\mathbb{E}_{\mathbf{x}}\left[\mathbf{x}\mathbf{x}^\top\mathbf{B}\mathbf{x}\mathbf{x}^\top\right] \preceq \mathcal{O}\left(\operatorname{tr}\left(\mathbf{H}^*\mathbf{B}\right)\mathbf{H}^*\right)$.*

Theorem 3.5 (Convergence Rate of PaZO on Least Squares)**.** *Suppose we are given access to the PSPSA, running PaZO for least squares regression* (6) *satisfying Assumption 3.4 with a learning rate* η *satisfying* $\frac{1}{\lambda_{min}((\mathbf{H}^*)^{1-2\alpha})T} \lesssim \eta \lesssim \min\left\{\frac{1}{\lambda_{max}((\mathbf{H}^*)^{1-2\alpha})}, \frac{\lambda_{\min}(\mathbf{H}^*)}{\lambda_{\max}(\mathbf{H}^*)\mathrm{tr}((\mathbf{H}^*)^{-2\alpha})\mathrm{tr}((\mathbf{H}^*)^{1-2\alpha})}\right\}$ *for* $2T$ *steps with* $T \gtrsim \frac{\lambda_{\max}(\mathbf{H}^*)\mathrm{tr}\left((\mathbf{H}^*)^{-2\alpha}\right)\mathrm{tr}\left((\mathbf{H}^*)^{1-2\alpha}\right)}{\lambda_{min}((\mathbf{H}^*)^{1-2\alpha})\lambda_{\min}(\mathbf{H}^*)}$ *allows PaZO to achieve the following convergence rate:*

$$\mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=T}^{2T-1}\boldsymbol{\theta}_t\right)\right] - f(\boldsymbol{\theta}^*) \leq \frac{\left(1-\eta\lambda_{\min}\left((\mathbf{H}^*)^{1-2\alpha}\right)\right)^T}{\eta T}\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}^*\|^2_{(\mathbf{H}^*)^{2\alpha}} + \frac{D_\alpha}{T}, \tag{8}$$

where $D_\alpha = \mathrm{tr}\left((\mathbf{H}^*)^{2\alpha-1}\right) \cdot \mathrm{tr}\left((\mathbf{H}^*)^{1-2\alpha}\right)$ *and* α *is the precondition order defined in PSPSA.*

Theorem 3.5 provides an affirmative answer to question *A*. Since the first term decays exponentially with T, the rate depends on the second term D_α/T, which is a trade-off between $\mathrm{tr}\left((\mathbf{H}^*)^{2\alpha-1}\right)$ and $\mathrm{tr}\left((\mathbf{H}^*)^{1-2\alpha}\right)$. Through Cauchy-Schwarz inequality, we have $D_\alpha \geq d^2$, where the equality holds if and only if $\alpha = 1/2$. In other words, only ZO-SGD is not enough to match the ideal rate $\tilde{\mathcal{O}}(d^2/T)$. Therefore, Theorem 3.5 demonstrates that different from first-order algorithms, *we need preconditions in zeroth-order optimization.*

Moreover, we consider the convergence analysis with approximate Hessian $\tilde{\mathbf{H}}_t$ in PSPSA. When the gap between $\tilde{\mathbf{H}}_t$ and $\mathbf{H}_t$ can be well controlled, we can also achieve the fastest rate when $\alpha = 1/2$. The detailed assumption and analysis are shown in Appendix B.

3.3 General Functions

Second, we propose the theoretical analysis for general smooth functions. Based on the affirmative answer to question *A* provided by Theorem 3.5, we conducted a more in-depth analysis of general functions, thereby establishing a more reasonable solution to question *B*. We may also obtain the results under approximate Hessian. For convenience, we assume its exact.

Assumption 3.6 (Gradient Uniform Continuity)**.** *For any given sample pair* $(\mathbf{x}, y) \sim \mathcal{D}$*, the stochastic gradient of the objective* $\nabla F(\boldsymbol{\theta}; (\mathbf{x}, y))$ *satisfies uniform continuity.*

Assumption 3.7 (General Hessian Smooth)**.** *For any given* $\boldsymbol{\theta}_1$*,* $\boldsymbol{\theta}_2$ *and* $\alpha' \in [-1, 1]$*, the Hessian of the objective* $\mathbf{H}(\boldsymbol{\theta}_1)$ *and* $\mathbf{H}(\boldsymbol{\theta}_2)$ *are invertible and satisfy*

$$\left\|\mathbf{H}^{\alpha'}(\boldsymbol{\theta}_1) - \mathbf{H}^{\alpha'}(\boldsymbol{\theta}_2)\right\| \leq \rho|\alpha'|\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|^{|\alpha'|}.$$

Assumption 3.7 is the generalization form of Lipschitz continuity of Hessian. When $\alpha = 1$, it reduces to Hessian Lipschitz continuity. We use it to limit the gap between $\mathbf{H}_t^{-2\alpha}$ in PSPSA and $(\mathbf{H}^*)^{-2\alpha}$. When the objective is strongly convex, the Hessian is naturally invertible, while for others we assume its invertible property. We propose the convergence rate of general functions in Theorem 3.8.

Theorem 3.8 (Convergence Rate of PaZO on General Functions)**.** *Suppose we are given access to the PSPSA, running PaZO for general functions* (1) *satisfying Assumption 3.3, 3.6 and 3.7 with a learning rate* η *satisfying* $\frac{1}{\lambda_{min}((\mathbf{H}^*)^{1-2\alpha})T} \lesssim \eta \leq \frac{1}{\lambda_{max}((\mathbf{H}^*)^{1-2\alpha})}$ *for* $2T$ *steps with* $T \gtrsim \frac{\lambda_{max}\left((\mathbf{H}^*)^{1-2\alpha}\right)}{\lambda_{min}((\mathbf{H}^*)^{1-2\alpha})}$ *where* $\mathbf{H}^*$ *is full-rank and* $\mathbb{E}\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|^p \leq \epsilon_0^p$ *for any* $t \in [T, 2T-1]$ *and* $p \in [0, 3]$ *allows PaZO to achieve the following asymptotic convergence rate:*

$$\begin{aligned}\mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=T}^{2T-1}\boldsymbol{\theta}_t\right)\right] - f(\boldsymbol{\theta}^*) \lesssim & \frac{(1-\eta\lambda_{\min}((\mathbf{H}^*)^{1-2\alpha}))^{2T}}{\eta^2T^2}\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}^*\|^2_{(\mathbf{H}^*)^{4\alpha-1}} \\ & + \frac{\mathrm{tr}\left((\mathbf{H}^*)^{2\alpha-1}\right)\cdot\mathrm{tr}\left((\mathbf{H}^*)^{1-2\alpha}\right)}{T} + \bar{\mathbf{Err}},\end{aligned} \tag{9}$$

where $\bar{\mathbf{Err}} = \mathcal{O}\left(\eta\rho\epsilon_0^3 + \eta\rho\epsilon_0^{2|\alpha|+1} + \eta^2\epsilon_0 + \eta^2\rho\epsilon_0^{2|\alpha|}\right)$ *represents the higher-order infinitesimal term, and* α *is the precondition order defined in PSPSA.*

We observe that the dominant term of the rate in Theorem 3.8 aligns with the rate in Theorem 3.5, which further demonstrates the generalized validity of our analysis on the role of preconditioning in zeroth-order optimization: for general problems, selecting $\alpha = 0$ alone induces a slower convergence

Algorithm 1 PaZO (Practical Form)

Require: parameters $\Theta = \{\boldsymbol{\theta}_i \in \mathbb{R}^{d_i}\}$, loss $\mathcal{L} : \mathbb{R}^d \to \mathbb{R}$, running steps T, perturbation scale μ, learning rate schedule η_t, smooth scale β_1, β_2, initialized diagonal Hessian $\boldsymbol{\Sigma}_0 = \mathbf{I}$, random seed s, a random number generator, Hessian reset frequency T_0

for $t = 1, ..., T$ **do**

  Step 1: Perturb Parameters through Diagonal Hessian

    Sample batch $\mathcal{B} \subset \mathcal{D}$ and random seed s

    $\ell \leftarrow \mathcal{L}(\boldsymbol{\theta}; \mathcal{B})$

    $\boldsymbol{\theta} \leftarrow$ PerturbParameters($\boldsymbol{\theta}$, μ, $\boldsymbol{\Sigma}_{t-1}^{-1/2}$, s)

    $\ell_+ \leftarrow \mathcal{L}(\boldsymbol{\theta}; \mathcal{B})$

    $\boldsymbol{\theta} \leftarrow$ PerturbParameters($\boldsymbol{\theta}$, -2μ, $\boldsymbol{\Sigma}_{t-1}^{-1/2}$, s)

    $\ell_- \leftarrow \mathcal{L}(\boldsymbol{\theta}; \mathcal{B})$

    $\boldsymbol{\theta} \leftarrow$ PerturbParameters($\boldsymbol{\theta}$, μ, $\boldsymbol{\Sigma}_{t-1}^{-1/2}$, s)   {Reset parameters before descent}

  Step 2: Estimate Diagonal Hessian

    $\tilde{\mathbf{g}} \leftarrow (\ell_+ - \ell_-) * \boldsymbol{\Sigma}_t^{1/2}\mathbf{u}/2\mu$ {Estimate unbiased zeroth-order gradient}

    $\tilde{\boldsymbol{\Sigma}} = \left((1-\beta_1)\boldsymbol{\Sigma}_{t-1}^2 + \beta_1 \cdot \mathrm{diag}(\tilde{\mathbf{g}} \circ \tilde{\mathbf{g}})\right)^{1/2}$ {Adding information from ZO gradient}

    $\boldsymbol{\Sigma}_t = \frac{1}{2\mu^2}(\ell_+ + \ell_- - 2\ell)\left(\tilde{\boldsymbol{\Sigma}}\left(\mathrm{diag}(\mathbf{u}\mathbf{u}^\top) - \mathbf{I}\right)\right)$ {Estimate unbiased diagonal Hessian}

  Step 3: Take Moving Average and Reset Diagonal Hessian

    $\boldsymbol{\Sigma}_t \leftarrow (1-\beta_2)\boldsymbol{\Sigma}_{t-1} + \beta_2|\boldsymbol{\Sigma}_t|$ {Take moving average of diagonal Hessian}

  if $t\%T_0 = 0$ **then**

    $\boldsymbol{\Sigma}_t \leftarrow \mathbf{I}$ {Frequently reset diagonal Hessian}

  end if

  Step 4: Update the Parameters

    Reset random number generator with seed s {For sampling $\mathbf{u}_i$}

    preconditioned_grad $\leftarrow (\ell_+ - \ell_-) * \boldsymbol{\Sigma}_t^{-1/2}/2\mu$ {Using $\boldsymbol{\Sigma}^{-1}$ as preconditioner}

  for $\boldsymbol{\theta}_i \in \Theta$ **do**

    Sample $\mathbf{u}_i \sim \mathcal{N}(0, \mathbf{I}_{d_i})$

    $\boldsymbol{\theta}_i \leftarrow \boldsymbol{\theta}_i - \eta_t *$ preconditioned_grad $* \mathbf{u}_i$

  end for

end for

Algorithm 2 PerturbParameters

Require: model parameters $\Theta = \{\boldsymbol{\theta}_i \in \mathbb{R}^{d_i}\}$, perturbation scale μ, diagonal Hessian $\boldsymbol{\Sigma}_t^{-1/2}$, random seed s, a random number generator

Reset random number generator with seed s {For sampling $\mathbf{u}_i$}

for $\boldsymbol{\theta}_i \in \Theta$ **do**

  Sample $\mathbf{u}_i \sim \mathcal{N}(0, \mathbf{I}_{d_i})$

  $\boldsymbol{\theta}_i \leftarrow \boldsymbol{\theta}_i + \mu\boldsymbol{\Sigma}_t^{-1/2}\mathbf{u}_i$ {Modify parameters in place}

end for

rate than the optimal choice $\alpha = 1/2$. Moreover, the $\bar{\mathbf{Err}}$ is defined as the higher-order infinitesimal term $\mathcal{O}\left(\eta\rho\epsilon_0^3 + \eta\rho\epsilon_0^{2|\alpha|+1} + \eta^2\epsilon_0 + \eta^2\rho\epsilon_0^{2|\alpha|}\right)$ that negligibly impacts the convergence rate of the dominant term. When T grows, we can choose the smaller η to obtain the controlled $\bar{\mathbf{Err}}$.Thus, Theorem 3.5 can be regarded as a special case of Theorem 3.8, and we propose the proof in detail in Appendix A. By selecting $\alpha = 1/2$, PaZO achieves the fastest convergence rate $\tilde{\mathcal{O}}\left(d^2/T\right)$ compared with MeZO and HiZOO for general functions, providing a reasonable answer to question *B*.

4 Algorithm for Fine-Tuning LLMs in Practice

In this section, we introduce PaZO (Practical Form) in Algorithm 1 for fine-tuning LLMs in practice. We provide an answer to the question below.

> *C.* How to effectively estimate Hessian information through zeroth-order perturbations in practice and improve fine-tuned model performance on downstream tasks?

Specifically, we apply the theoretically optimal order of preconditioner $\mathbf{H}^{-1/2}$ in the PSPSA process. Then we estimate diagonal Hessian with incorporating the current zeroth-order gradient information and moving average techniques through the same PSPSA process for estimating the preconditioned zeroth-order gradient. Our algorithm can be divided into four steps.

Step I. Perturb Parameters through Diagonal Hessian. First, we apply PSPSA to our practical algorithm to obtain the preconditioned zeroth-order gradient. Inspired by our theoretical results, we use $\mathbf{\Sigma}^{-1/2}$ as the preconditioner in the PSPSA process, where $\mathbf{\Sigma}$ is the estimated diagonal Hessian. Through twice forward passes of PSPSA we obtain

$$\ell_+ = F(\boldsymbol{\theta} + \mu\mathbf{\Sigma}^{-1/2}\mathbf{u}; (\mathbf{x}, y)), \;\; \ell_- = F(\boldsymbol{\theta} - \mu\mathbf{\Sigma}^{-1/2}\mathbf{u}; (\mathbf{x}, y)).$$

Moreover, we run another additional forward pass before adding perturbation to obtain $\ell = F(\boldsymbol{\theta}; (\mathbf{x}, y))$ for estimating $\mathbf{\Sigma}$ in the following steps.

Step II. Estimate Diagonal Hessian. We try to estimate the diagonal Hessian through ℓ_+, ℓ_- and ℓ, with $\mathcal{O}(d)$ memory cost against $\mathcal{O}(d^2)$ for the full Hessian. Specifically, in the theoretical analysis of the Hessian-aware zeroth-order optimization [55], they demonstrate that

$$\mathbb{E}_{\mathbf{u}\sim\mathcal{N}(0,\mathbf{I}_d)}\left[\frac{1}{2}\mathbf{u}^\top\mathbf{A}^{\frac{1}{2}}\mathbf{H}\mathbf{A}^{\frac{1}{2}}\mathbf{u}\cdot\left(\mathbf{A}^{-\frac{1}{2}}\mathbf{u}\mathbf{u}^\top\mathbf{A}^{-\frac{1}{2}} - \mathbf{A}^{-1}\right)\right] = \mathbf{H}, \tag{10}$$

where $\mathbf{H}$ is the Hessian matrix, and $\mathbf{A}$ is *any* given positive definite matrix. Thus, letting $\mathbf{\Sigma}$ be a positive definite diagonal matrix and setting $\mathbf{A} = \mathbf{\Sigma}^{-1}$, we obtain the diagonal version of (10) as

$$\mathbb{E}\left[\underbrace{\frac{1}{2}\mathbf{u}^\top\mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{H}\mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{u}}_{\mathcal{I}}\cdot\mathbf{\Sigma}\left(\mathrm{diag}(\mathbf{u}\mathbf{u}^\top) - \mathbf{I}\right)\right] = \mathbf{H}. \tag{11}$$

We use ℓ_+, ℓ_- and ℓ to estimate $\mathcal{I}$. Through Talyor expansion, we have

$$\begin{aligned}\ell_+ &= F(\boldsymbol{\theta}; (\mathbf{x}, y)) + \mu\left\langle\nabla F(\boldsymbol{\theta}; (\mathbf{x}, y)), \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{u}\right\rangle + \frac{\mu^2}{2}\mathcal{I} + \mathcal{O}(\mu^3),\\ \ell_- &= F(\boldsymbol{\theta}; (\mathbf{x}, y)) - \mu\left\langle\nabla F(\boldsymbol{\theta}; (\mathbf{x}, y)), \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{u}\right\rangle + \frac{\mu^2}{2}\mathcal{I} + \mathcal{O}(\mu^3).\end{aligned} \tag{12}$$

Thus, we can obtain $\mathcal{I}$ by the combination of ℓ_+, ℓ_- and ℓ as

$$\frac{\ell_+ + \ell_- - \ell}{\mu^2} = \mathcal{I} + \mathcal{O}(\mu). \tag{13}$$

Moreover, incorporating the current gradient information into the preconditioner is demonstrated to be effective in first-order optimizers [28, 35].We additional estimate

$$\tilde{\mathbf{g}} = (\ell_+ - \ell_-) * \frac{\mathbf{\Sigma}_t^{1/2}\mathbf{u}}{2\mu} = \mathbf{u}\mathbf{u}^\top\nabla F(\boldsymbol{\theta}; (\mathbf{x}, y)) + \mathcal{O}(\mu)$$

as an unbiased zeroth-order gradient and incorporate $\mathrm{diag}(\tilde{\mathbf{g}} \circ \tilde{\mathbf{g}})$ as a correction item to integrate local first-order estimated information into $\mathbf{\Sigma}_t$ through a moving average mechanism as

$$\tilde{\mathbf{\Sigma}} = \left((1-\beta_1)\mathbf{\Sigma}_{t-1}^2 + \beta_1\cdot\mathrm{diag}(\tilde{\mathbf{g}} \circ \tilde{\mathbf{g}})\right)^{1/2}.$$

Then we use (11) to update the diagonal Hessian as

$$\mathbf{\Sigma}_t = \frac{1}{2\mu^2}(\ell_+ + \ell_- - 2\ell)\left(\tilde{\mathbf{\Sigma}}\left(\mathrm{diag}(\mathbf{u}\mathbf{u}^\top) - \mathbf{I}\right)\right). \tag{14}$$

Step III. Take Moving Average and Reset Diagonal Hessian. In practice, we empirically discover the instability of the estimated diagonal Hessian. To solve this problem, we take the moving average of the historical estimate and the current one to maintain the smoothness and stability of $\mathbf{\Sigma}_t$ as

$$\mathbf{\Sigma}_t = (1-\beta_2)\mathbf{\Sigma}_{t-1} + \beta_2|\mathbf{\Sigma}_t|, \tag{15}$$

Table 1: Experiments on RoBERTa-large (350M parameters, k=16). We use zero-shot learning, linear probing (LP), full-parameter fine-tuning with Adam, MeZO and PaZO on six downstream tasks. We also test PEFT methods including LoRA and prefix tuning with Adam, MeZO and PaZO respectively. All reported numbers are averaged accuracy (standard deviation) across 5 runs.

Task Type	**SST-2**	**SST-5**	**SNLI**	**MNLI**	**RTE**	**TREC**	**Average**
	—— sentiment ——		—— natural language inference ——			— topic —	
Zero-shot	79.0	35.5	50.2	48.8	51.4	32.0	49.5
LP	76.0 (±2.8)	40.3 (±1.9)	66.0 (±2.7)	56.5 (±2.5)	59.4 (±5.3)	51.3 (±5.5)	58.3
FT	90.9 (±1.7)	44.8 (±1.6)	67.5(±2.4)	58.2 (±3.1)	66.4 (±7.2)	85.0 (±2.5)	68.8
FT (PEFT)	91.9 (±1.0)	43.2 (±1.1)	65.5 (±1.8)	57.1 (±1.3)	65.5 (±1.9)	79.8 (±1.5)	67.2
MeZO	90.5 (±1.2)	42.3 (±2.1)	66.7 (±3.3)	51.6 (±3.0)	**64.0** (±3.3)	70.2 (±1.4)	64.2
MeZO (PEFT)	91.3 (±1.0)	42.4 (±2.5)	62.7 (±2.8)	55.6 (±2.0)	60.5 (±3.6)	73.4 (±3.6)	64.3
PaZO	**91.4** (±0.8)	**44.6** (±1.7)	**66.7** (±2.6)	**56.4** (±2.1)	63.2 (±5.2)	70.8 (±2.0)	**65.6**
PaZO (PEFT)	91.3 (±0.3)	42.9 (±0.5)	62.4 (±1.6)	55.8 (±1.7)	61.5 (±2.2)	**77.4** (±3.5)	65.2

where $|\mathbf{\Sigma}_t|$ means taking the absolute values of $\mathbf{\Sigma}_t$ to maintain positive definite. Moreover, when the iteration step exceeds a threshold, excessive accumulated historical information may no longer positively contribute. Therefore, we reset the $\mathbf{\Sigma}$ frequently after some steps.

Step IV. Update the Parameters. Finally, we layer-wisely compute the preconditioned gradient by PSPSA, where the gradient estimate is equivalent to a $\mathbf{\Sigma}^{-1}$ preconditioned zeroth-order gradient.

5 Experiment

We conduct experiments on both masked LMs (RoBERTa-large, 350M [34]) and large-scale generative LMs (OPT-1.3B [57]) with zero-shot learning, linear probing (LP [22]), in-context learning (ICL [8]), full-parameter tuning and PEFT including LoRA [23] and prefix-tuning [31] (see Appendix C.3 for details). We compare PaZO with other representative zeroth-order optimizers including MeZO and HiZOO (see Appendix C.4 for details). We first show that PaZO achieves significant improvement over zero-shot, ICL, and LP. Compared with first-order optimizers (FT), PaZO drastically reduces the memory cost while maintaining comparable performance. Moreover, PaZO realizes better performance compared with MeZO and HiZOO. Detailed settings are presented in Appendix C.2.

5.1 Masked Language Models

We conduct experiments for RoBERTa-large (350M) on sentiment classification, natural language inference, and topic classification tasks. We sample k examples per class for $k = 16$, running zeroth-shot learning, LP, fine-tuning, MeZO and PaZO. We summarize the results in Table 1. First, we show that: (1) PaZO works significantly better than zero-shot and LP; (2) PaZO achieves comparable performance to FT. Moreover, we show the better performance of PaZO compared with MeZO.

PaZO achieves better performance compared with MeZO. As shown in Table 1, PaZO achieves improved performance on average across all the datasets, tasks and PEFT (we choose the best results from LoRA and prefix-tuning). For sentiment tasks, the improvement of PaZO is universal, while for NLI and topic tasks the improvement is evident on MNLI and TREC with 9.3% and 5.4%.

5.2 Generative Language Models

We extend the experiments to the OPT 1.3B model [57] on classification and multiple-choice tasks on different datasets (see Appendix C.1 for details). We randomly sample 1000, 500, and 1000 examples for training, validation, and test sets, respectively, for each dataset. We run MeZO, HiZOO and PaZO for 20K steps, and compare the performance with different zeroth-order optimizers in Table 2.

PaZO achieves SOTA performance compared with other zeroth-order optimizers. As shown in Table 2, PaZO achieves SOTA performance compared to other zeroth-order optimizer baselines including MeZO ($\alpha = 0$) and HiZOO ($\alpha = -1/2$). Specifically, for average performance, PaZO achieves all-round improvement beyond MeZO and HiZOO, no matter the full-parameter version, the

Table 2: Performance comparison with MeZO and HiZOO. We fine-tune OPT-1.3B on different downstream datasets and evaluate the performance, applying LoRA and prefix-tuning.

Task Type	SST-2	BoolQ	CB	ReCoRD	RTE	WIC	WSC	COPA	MultiRC	Average
	——————- classification ——————-							– multiple choice –		
MeZO	88.5	63.4	67.8	72.3	66.1	60.6	57.6	76.0	56.3	67.6
MeZO (LoRA)	88.5	63.0	60.7	70.6	59.9	58.2	54.8	77.0	58.9	65.7
MeZO (prefix)	91.3	**64.1**	67.9	71.0	62.5	54.2	51.2	75.0	57.2	66.0
HiZOO	88.5	61.4	67.9	71.9	64.3	62.2	**62.5**	73.0	**59.3**	67.9
HiZOO (LoRA)	88.5	63.1	69.6	**72.5**	64.6	60.6	54.8	76.0	58.9	67.6
HiZOO (prefix)	91.3	63.6	67.9	70.9	63.2	53.8	57.7	75.0	54.5	66.4
PaZO	89.0	63.4	69.6	72.1	**66.4**	**63.2**	61.5	75.0	57.6	**68.6**
PaZO (LoRA)	88.5	63.4	**73.2**	72.1	62.8	58.2	54.8	**77.0**	58.9	67.7
PaZO (prefix)	**91.3**	63.4	67.9	71.0	62.3	53.8	57.7	75.0	57.2	66.6

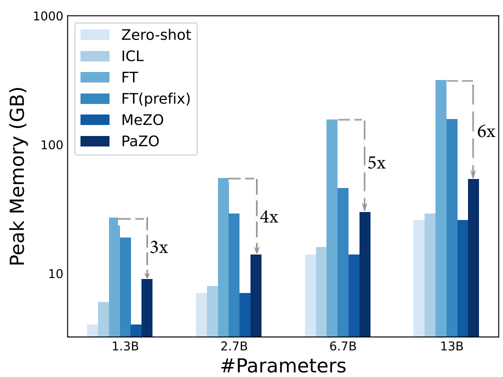

Figure 1: GPU peak memory overhead with different OPT models and tuning methods on MultiRC (400 tokens per example on average). See Appendix C.5 for details.

	MeZO	HiZOO	PaZO
RoBERTa-L	0.2091s	0.3020s	0.3046s
RoBERTa-L1	0.1338s	0.1993s	0.2013s
RoBERTa-L2	0.1254s	0.1869s	0.1892s
OPT-1.3B	0.2564s	0.3812s	0.3837s
OPT-1.3B1	0.1664s	0.2798s	0.2857s
OPT-1.3B2	0.1572s	0.2374s	0.2419s

Figure 2: Wallclock time per step among MeZO, HiZOO and PaZO. The increase in wallclock time per step for PaZO compared to MeZO is less than 1.5 times across different model sizes. All results are measured on the same dataset (SST-2) and GPUs (24GB 3090), with each result averaged over 100 steps.

LoRA version or the prefix-tuning version. For single-task performance, PaZO and its peft version show advantages in the vast majority of tasks and have little gaps in other tasks.

5.3 Memory Usage and Wall-clock Time Analysis

Memory Usage. As shown in Figure 1, PaZO has more memory overhead compared to MeZO because of the storage of the diagonal Hessian, and maintains the memory overhead compared to HiZOO. However, PaZO also exhibits extreme saving of memory compared to first-order optimizers, specifically, up to $6\times$ compared to standard FT and $3\times$ compared to FT (prefix-tuning).

Wall-clock Time. As shown in Figure 2, PaZO spends $1.5\times$ time per step compared with MeZO, and the same time per step compared with HiZOO, since preconditioned optimizers need an additional forward pass for estimating diagonal Hessian. In Figure 2, Model1 means we use LoRA and Model2 means we use prefix-tuning. Considering the accelerated convergence rate of PaZO with fewer steps to obtain the same loss, PaZO achieves better performance with acceptable extra time cost.

6 Conclusion

In this work, we propose PaZO, a preconditioned accelerated zeroth-order optimization method for fine-tuning LLMs. We theoretically analyze the necessity of preconditions in ZO, and demonstrate the optimal order of preconditioners to achieve the fastest convergence rate. We propose the practical form of PaZO and extensive experiments on different models and tasks show the effectiveness.

References

[1] Josh Achiam et al. "Gpt-4 technical report". In: *arXiv preprint arXiv:2303.08774* (2023).

[2] Alekh Agarwal, Ofer Dekel, and Lin Xiao. "Optimal algorithms for online convex optimization with multi-point bandit feedback." In: *Colt*. Citeseer. 2010, pp. 28–40.

[3] Alekh Agarwal et al. "Information-theoretic lower bounds on the oracle complexity of convex optimization". In: *Advances in Neural Information Processing Systems* 22 (2009).

[4] Shun-ichi Amari et al. "When does preconditioning help or hurt generalization?" In: *arXiv preprint arXiv:2006.10732* (2020).

[5] Francis Bach and Eric Moulines. "Non-strongly-convex smooth stochastic approximation with convergence rate O (1/n)". In: *Advances in neural information processing systems* 26 (2013).

[6] Luisa Bentivogli et al. "The Fifth PASCAL Recognizing Textual Entailment Challenge." In: *TAC* 7.8 (2009), p. 1.

[7] Samuel R Bowman et al. "A large annotated corpus for learning natural language inference". In: *arXiv preprint arXiv:1508.05326* (2015).

[8] Tom Brown et al. "Language models are few-shot learners". In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[9] Charles George Broyden. "The convergence of a class of double-rank minimization algorithms 1. general considerations". In: *IMA Journal of Applied Mathematics* 6.1 (1970), pp. 76–90.

[10] HanQin Cai et al. "A zeroth-order block coordinate descent algorithm for huge-scale black-box optimization". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 1193–1203.

[11] Pin-Yu Chen et al. "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*. 2017, pp. 15–26.

[12] Yiming Chen et al. "Enhancing zeroth-order fine-tuning for language models with low-rank structures". In: *arXiv preprint arXiv:2410.07698* (2024).

[13] Christopher Clark et al. "Boolq: Exploring the surprising difficulty of natural yes/no questions". In: *arXiv preprint arXiv:1905.10044* (2019).

[14] Ido Dagan, Oren Glickman, and Bernardo Magnini. "The pascal recognising textual entailment challenge". In: *Machine learning challenges workshop*. Springer. 2005, pp. 177–190.

[15] Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. "The commitment-bank: Investigating projection in naturally occurring discourse". In: *proceedings of Sinn und Bedeutung*. Vol. 23. 2. 2019, pp. 107–124.

[16] John C Duchi et al. "Optimal rates for zero-order convex optimization: The power of two function evaluations". In: *IEEE Transactions on Information Theory* 61.5 (2015), pp. 2788–2806.

[17] Rong Ge et al. "The step decay schedule: A near optimal, geometrically decaying learning rate procedure for least squares". In: *Advances in neural information processing systems* 32 (2019).

[18] Danilo Giampiccolo et al. "The third pascal recognizing textual entailment challenge". In: *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*. 2007, pp. 1–9.

[19] Suchin Gururangan et al. "Don't stop pretraining: Adapt language models to domains and tasks". In: *arXiv preprint arXiv:2004.10964* (2020).

[20] R Bar Haim et al. "The second pascal recognising textual entailment challenge". In: *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*. Vol. 7. 2006, pp. 785–794.

[21] Davood Hajinezhad and Michael M Zavlanos. "Gradient-free multi-agent nonconvex nonsmooth optimization". In: *2018 IEEE Conference on Decision and Control (CDC)*. IEEE. 2018, pp. 4939–4944.

[22] Kaiming He et al. "Masked autoencoders are scalable vision learners". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 16000–16009.

[23] Edward J Hu et al. "Lora: Low-rank adaptation of large language models." In: *ICLR* 1.2 (2022), p. 3.

[24] Prateek Jain et al. "Accelerating stochastic gradient descent for least squares regression". In: *Conference On Learning Theory*. PMLR. 2018, pp. 545–604.

[25] Prateek Jain et al. “Parallelizing stochastic approximation through mini-batching and tail-averaging”. In: *arXiv preprint arXiv:1610.03774* (2016).

[26] Kevin G Jamieson, Robert Nowak, and Ben Recht. “Query complexity of derivative-free optimization”. In: *Advances in Neural Information Processing Systems* 25 (2012).

[27] Daniel Khashabi et al. “Looking beyond the surface: A challenge set for reading comprehension over multiple sentences”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 2018, pp. 252–262.

[28] Diederik P Kingma. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[29] Brian Lester, Rami Al-Rfou, and Noah Constant. “The power of scale for parameter-efficient prompt tuning”. In: *arXiv preprint arXiv:2104.08691* (2021).

[30] Hector J Levesque, Ernest Davis, and Leora Morgenstern. “The Winograd schema challenge.” In: *KR* 2012 (2012), 13th.

[31] Xiang Lisa Li and Percy Liang. “Prefix-tuning: Optimizing continuous prompts for generation”. In: *arXiv preprint arXiv:2101.00190* (2021).

[32] Hong Liu et al. “Sophia: A scalable stochastic second-order optimizer for language model pre-training”. In: *arXiv preprint arXiv:2305.14342* (2023).

[33] Sijia Liu et al. “signSGD via zeroth-order oracle”. In: *International conference on learning representations*. 2019.

[34] Yinhan Liu et al. “Roberta: A robustly optimized bert pretraining approach”. In: *arXiv preprint arXiv:1907.11692* (2019).

[35] Ilya Loshchilov and Frank Hutter. “Decoupled weight decay regularization”. In: *arXiv preprint arXiv:1711.05101* (2017).

[36] George D. Magoulas, Michael N. Vrahatis, and George S Androulakis. “Improving the convergence of the backpropagation algorithm using learning rate adaptation methods”. In: *Neural Computation* 11.7 (1999), pp. 1769–1796.

[37] Sadhika Malladi et al. “Fine-tuning language models with just forward passes”. In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 53038–53075.

[38] James Martens et al. “Deep learning via hessian-free optimization.” In: *Icml*. Vol. 27. 2010, pp. 735–742.

[39] Arkadij Semenovič Nemirovskij and David Borisovich Yudin. “Problem complexity and method efficiency in optimization”. In: (1983).

[40] Yurii Nesterov and Boris T Polyak. “Cubic regularization of Newton method and its global performance”. In: *Mathematical programming* 108.1 (2006), pp. 177–205.

[41] Yurii Nesterov and Vladimir Spokoiny. “Random gradient-free minimization of convex functions”. In: *Foundations of Computational Mathematics* 17.2 (2017), pp. 527–566.

[42] Long Ouyang et al. “Training language models to follow instructions with human feedback”. In: *Advances in neural information processing systems* 35 (2022), pp. 27730–27744.

[43] Mohammad Taher Pilehvar and Jose Camacho-Collados. “WiC: the word-in-context dataset for evaluating context-sensitive meaning representations”. In: *arXiv preprint arXiv:1808.09121* (2018).

[44] Maxim Raginsky and Alexander Rakhlin. “Information-based complexity, feedback and dynamics in convex programming”. In: *IEEE Transactions on Information Theory* 57.10 (2011), pp. 7036–7056.

[45] Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. “Choice of Plausible Alternatives: An Evaluation of Commonsense Causal Reasoning.” In: *AAAI spring symposium: logical formalizations of commonsense reasoning*. 2011, pp. 90–95.

[46] Ohad Shamir. “An optimal algorithm for bandit and zero-order convex optimization with two-point feedback”. In: *Journal of Machine Learning Research* 18.52 (2017), pp. 1–11.

[47] Richard Socher et al. “Recursive deep models for semantic compositionality over a sentiment treebank”. In: *Proceedings of the 2013 conference on empirical methods in natural language processing*. 2013, pp. 1631–1642.

[48] James C Spall. “Multivariate stochastic approximation using a simultaneous perturbation gradient approximation”. In: *IEEE transactions on automatic control* 37.3 (1992), pp. 332–341.

[49] Yan Sun et al. “TeZO: Empowering the Low-Rankness on the Temporal Dimension in the Zeroth-Order Optimization for Fine-tuning LLMs”. In: *arXiv preprint arXiv:2501.19057* (2025).

[50] Yujie Tang, Junshan Zhang, and Na Li. “Distributed zero-order algorithms for nonconvex multiagent optimization”. In: *IEEE Transactions on Control of Network Systems* 8.1 (2020), pp. 269–281.

[51] Ellen M Voorhees and Dawn M Tice. “Building a question answering test collection”. In: *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*. 2000, pp. 200–207.

[52] Alex Wang et al. “Superglue: A stickier benchmark for general-purpose language understanding systems”. In: *Advances in neural information processing systems* 32 (2019).

[53] Zhongruo Wang et al. “Zeroth-order algorithms for nonconvex minimax problems with improved complexities”. In: *arXiv preprint arXiv:2001.07819* (2020).

[54] Adina Williams, Nikita Nangia, and Samuel R Bowman. “A broad-coverage challenge corpus for sentence understanding through inference”. In: *arXiv preprint arXiv:1704.05426* (2017).

[55] Haishan Ye. “Mirror natural evolution strategies”. In: *arXiv preprint arXiv:2308.00469* (2023).

[56] Sheng Zhang et al. “Record: Bridging the gap between human and machine commonsense reading comprehension”. In: *arXiv preprint arXiv:1810.12885* (2018).

[57] Susan Zhang et al. “Opt: Open pre-trained transformer language models”. In: *arXiv preprint arXiv:2205.01068* (2022).

[58] Yihua Zhang et al. “Revisiting zeroth-order optimization for memory-efficient llm fine-tuning: A benchmark”. In: *arXiv preprint arXiv:2402.11592* (2024).

[59] Yanjun Zhao et al. “Second-order fine-tuning without pain for llms: A hessian informed zeroth-order optimizer”. In: *arXiv preprint arXiv:2402.15173* (2024).

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: In the abstract, we briefly introduce our contribution while in the Introduction we propose the three problems we focus on and our three contributions followed behind "Our contributions are".

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: In the experiment part we compare the per-step time cost and analyze the reason our method is slower than the baseline per-step due to an additional forward pass.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: We provide all the assumptions in the "Theoretical Insights of PaZO" section, and provide a complete proof in Appendix A.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide the details of our experiments in Appendix C, including the models, dataset, methods and hyperparameters we use for reproducibility.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We are organizing our code and will make it public after the organization is completed.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We provide the details of our experiments in Appendix C.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We report the average results with the form ($\pm$std) in Table 1.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: In Appendix C.5, we provide the compute resources to reproduce the experiments with different scales and methods.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our research is with the NeurIPS Code of Ethics in every respect.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss that our algorithm can save memory cost when fine-tuning LLMs.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have cited the paper that produced the models and dataset we use.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We will make our algorithms and fine-tuned models public after the organization is completed with formal documents and codes.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [Yes]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: The aim of our research is to efficiently fine-tune pre-trained LLMs with lower memory cost, with faster speed compared with other zeroth-order methods.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.