Differentially Private Zeroth-Order Methods for Scalable Large Language Model Finetuning

Zhihao Liu
Zhejiang University
zhihao_liu@zju.edu.cn

Jian Lou
Zhejiang University
jian.lou@zju.edu.cn

Wenjie Bao
Zhejiang University
wenjie_bao@zju.edu.cn

Zhan Qin
Zhejiang University
qinzhan@zju.edu.cn

Kui Ren
Zhejiang University
kuiren@zju.edu.cn

Abstract

Finetuning on task-specific datasets is a widely-embraced paradigm of harnessing the powerful capability of pretrained LLMs for various downstream tasks. Due to the popularity of LLMs finetuning and its accompanying privacy concerns, differentially private (DP) finetuning of pretrained LLMs has garnered increasing attention to safeguarding the privacy of task-specific datasets. Lying at the design core of DP LLM finetuning methods is the satisfactory tradeoff between privacy, utility, and scalability. Most existing methods build upon the seminal work of DP-SGD. Despite pushing the scalability of DP-SGD to its limit, DP-SGD-based finetuning methods are unfortunately limited by the inherent inefficiency of SGD.

In this paper, we investigate the potential of DP zeroth-order methods for LLM pretraining, which avoids the scalability bottleneck of SGD by approximating the gradient with the more efficient zeroth-order gradient. Rather than treating the zeroth-order method as a drop-in replacement for SGD, this paper presents a comprehensive study both theoretically and empirically. First, we propose the stagewise DP zeroth-order method that dynamically schedules key hyperparameters. This design is grounded on the synergy between DP random perturbation and the gradient approximation error of the zeroth-order method, and its effect on finetuning trajectory. Second, we further enhance the scalability by reducing the trainable parameters that are identified by repurposing a data-free pruning technique requiring no additional data or extra privacy budget. We provide theoretical analysis for both proposed methods. We conduct extensive empirical analysis on both encoder-only masked language model and decoder-only autoregressive language model, achieving impressive results in terms of scalability and utility.

1 Introduction

Pretrained Large Language Models (LLMs), scaling up to unprecedented sizes, have demonstrated remarkable potential in their capabilities of understanding and processing natural languages with human-like proficiency [31] [32]. This has prompted a rapid surge in demand to harness the power of pretrained LLMs, particularly open-sourced series like OPT [52], llama [39] and GPT [32], to boost performance across a wide range of downstream tasks. Finetuning is one of the most fundamental and dominant approaches for adapting pretrained LLMs to specific downstream tasks, which has been proven effective in [53] [36]. Finetuning starts with the publicly accessible checkpoint of the selected model and continues to train the model for several epochs based on the task-specific dataset (referred to as the finetuning dataset hereafter). While appearing a simple process at first glance, the sheer scale of nowadays LLMs introduces significant scalability issues for finetuning, e.g., incurring prohibitive memory footprint, which complicates the design of training methods even in nonprivate finetuning [50].

It is widely recognized that a finetuned LLM can leak sensitive information [5] [6] from its finetuning dataset, which is considered private and valuable in certain downstream application areas related to finance, healthcare [6]. Therefore, private finetuning of pretrained LLMs has become a pressing need due to the escalating privacy concerns associated with the growing popularity of LLMs [46]. Differential privacy (DP) [12] is a widely adopted mathematical framework that ensures a rigorous privacy protection guarantee by introducing calibrated random perturbations. A long-standing research theme in DP is the pursuit of an ideal tradeoff between privacy and utility, which arises because the random perturbation for privacy-preserving purpose will inevitably degrade the utility. DP LLM fine-tuning faces even intensified tension as it has to handle the tradeoff among privacy, utility, and scalability.

Most of the existing DP LLM finetuning methods [23] [19] build upon the seminal work of Differentially Private SGD (DP-SGD) [1], which clips the gradient vector per training sample, injects random Gaussian noise after clipping, and tracks the privacy budget across iterations by the moments accountant technique. These DP-SGD-based methods mitigate the scalability issue in roughly two lines of effort. The first line [25] [3] [19] aims to reduce the computational cost of the per-sample clipping of the gradient vector that is known to incur heavy computational and memory burden. For instance, [25] introduce group clipping to replace the per-sample clip, [4] propose book keep (BK) strategy. The second line [15] incorporates DP-SGD with parameter-efficient finetuning techniques in LLMs, which limits the amount of trainable parameters. The intuition is that the utility-privacy tradeoff degrades quickly with respect to the number of trainable parameters since the DP perturbations need to be injected into all dimensions of the gradient vector. For instance, [46] [23] consider LoRA, adapters, and XX, which either train two smaller factors to approximate the LLM parameter update or insert additional trainable modules while freezing the pretrained LLM parameters.

Although the DP-SGD-based methods mentioned above have made the best effort to squeeze performance out of DP-SGD, the inefficiency inherent in SGD continues to hinder achieving a satisfactory "privacy-utility-scalability" tradeoff for LLM finetuning. That is, the gradient calculation requires caching all intermediate activations during the forward pass and gradients during the backward pass, which leads to prohibitive memory usage for LLM finetuning, i.e., consuming up to 12× the memory usage required for inference [27]. Driven by such limitation, recent nonprivate LLM finetuning methods turn to zeroth-order methods, which circumvent the scalability issues associated with gradient computations by approximating them using two inferences. In nonprivate LLM finetuning, the effectiveness of zeroth-order methods has been validated through both theoretical and empirical studies, which suggests a promising direction for developing the DP LLM finetuning method.

In this paper, we strive to achieve a better "privacy-utility-scalability" tradeoff for DP LLM finetuning, through the lens of zeroth-order methods that are previously under-explored in DP literature. We comprehensively investigate DP zeroth-order methods from both theoretical and empirical perspectives, rather than treating the zeroth-order method as a drop-in replacement for SGD.

Our contributions. First, we focus on the synergy between the DP random perturbations and the gradient approximation error of Zeroth-order estimation to the true gradient, across different training stages. Specifically, we propose to dynamically adjust the ZO scale parameter that controls the gradient approximation error and divide the DP LLM finetuning into stages. In earlier stages, the gradient approximation error is controlled to be smaller, and together with a larger learning rate, it allows the LLM finetuning to more quickly approach the optimum. In the later stages, we need to deliberately increase the gradient approximation error, together with a decreased learning rate, offering a stabilization effect on the finetuning trajectory. Our theoretical analysis demonstrates that this stagewise DP Zeroth-order finetuning strategy provides an improved convergence rate.

Second, we explore the proposed stagewise DP Zeroth-order method in conjunction with reduced trainable parameters. Unlike existing works in DP-SGD, which introduce additional trainable modules to modify the LLM structure, or the LoRA-based method, which still entails a certain number of variables, we propose to initially identify key parameters within the given LLM. Subsequently, we treat these identified parameters as trainable while freezing the remainder. To identify the key parameters, we repurpose a data-free pruning method, which does not necessitate public data or incur additional privacy costs during the pruning stage. This advanced DP stagewise Zeroth-order method with pruning also comes with theoretical analysis.

Third, we conduct extensive empirical evaluations to corroborate the superiority of our proposed DP Zeroth-order methods for LLM finetuning. We utilize four different open-source LLMs, including masked language model (Roberta-large), autoregressive language model (OPT-2.7B), LLaMA-7B (to be completed) and GPT-NEO-1.3B (to be completed) for downstream tasks such as sentiment classification, natural language inference, and topic classification (including datasets like SST-2, SST-5, SNLI, MNLI and TREC). Our method exhibits strong scalability and performs well across all models and datasets. Our method has enhanced the model's performance greatly (about 4.2% on TREC when $\varepsilon = 8$) compared to existing solutions.

Our main contributions can be summarized as follows:

- We are the first to conduct a comprehensive investigation of DP LLM fine-tuning from a zeroth-order perspective.
- We introduce two novel stagewise DP Zeroth-order methods and provide a comprehensive theoretical analysis of privacy and convergence for both of them.
- We extensively experiment with our methods, providing empirical evidence of their scalability and utility.

Remarks on Two Concurrent Works. While finalizing our paper, we became aware of two concurrent yet preliminary works that also explore differentially private zeroth-order methods: a workshop paper [49] and a "work in progress" paper on arXiv [38]. We stress that our work differs from them in three key aspects: 1) Regarding algorithm design, all these works merely consider constant scheduling of hyperparameters and have not explored parameter sparsification to further boost scalability like us; 2) Regarding theoretical analysis: [38] did not provide any theoretical analysis for the utility. [49]'s utility analysis did not consider the dynamic scheduling of the hyperparameters that render the analysis more involved; 3) Regarding empirical analysis: [49] did not provide any empirical studies. While [38] also considered LLM finetuing, our empirical study is more comprehensive by considering both RoBERTa-large and OPT models, along with classification and multiple tasks.

In sum, we are the only work providing both theoretical and empirical studies among all concurrent works, offering more involved theoretical analysis and more comprehensive empirical analysis. Therefore, we believe our work offers distinct and significant contributions to both fields of DP LLM fine-tuning and DP zeroth-order methods, compared to these concurrent works.

2 Background and Related Work

2.1 Private Parameter Efficient Fine Tuning

As LLMs scale up, full parameter fine-tuning costs huge GPU memory. Parameter-efficient fine-tuning methods reduce memory consumption by updating just a fraction of the network parameters. Differentially private prompt tuning [24] [11] has emerged as a simple and parameter-efficient solution for steering LLMs to downstream tasks. DP fine-tuning with Reparametrized Gradient Perturbation [47] [23] computes the low-dimension projected gradient without computing the gradient itself, significantly reducing computation cost.

2.2 DP Fine-Tuning without Backpropagation

Fine-tuning methods without explicit gradient information decrease memory consumption by replacing backpropagation with inferences. Private in-context learning [29] involves task adaptation without modifying a pre-trained model's weights. ICL concatenates a sequence of k exemplars (query-answer pairs) before the questions, and then queries the model. However, ICL does not improve model performance on downstream tasks effectively. Zeroth-order method can estimate approximate gradients using only two forward passes, reducing memory consumption while achieving better performance than ICL [27]. However, differentially private zeroth-order method still lacks comprehensive and in-depth research.

2.3 Stagewise Training Strategy

Stagewise training strategy is widely used to solve weakly-convex problems [18] [8]. Stagewise algorithm starts with a relatively large learning rate and geometrically decrease the step size after several iterations. Stagewise training strategy are proven to have faster convergence rate theoretically in both first-order method [48] and zeroth-order method [30].

2.4 Pruning

Pruning methods like SNIP (Sparse Neural Network Pruning) [22] and GraSP (Gradient Signal Preservation) [43] are often used to identify and remove unimportant connections within the network to preserve the gradient flow after pruning. Zeroth-order pruning method (ZO-GraSP) [7] is proposed to identify high-quality sparse subnetwork via ZO oracle. However, all the above pruning methods are data-dependent which can be hard to satisfy in private settings. Tanaka et al. [37] proposed data-free pruning method SynFlow. Data-free SynFlow metric is leveraged to guide the selective removal of less critical parameters from neural networks, resulting in streamlined models without significant loss of performance.

3 Preliminary

3.1 LLM Fine-tuning

LLM fine-tuning has been a popular way of adapting a pre-trained large language model to a specific task or domain by further training it on task-specific data.

Definition 1 (LLM Fine-tuning) . *Fine-tuning a pretained language model $f(\theta)$ on the dataset $\mathcal{D}$ of downstream task can be described as the following optimization problem:*

$$\min_{\theta \in \mathcal{R}^d} \{f(\theta, \mathcal{D})\} := \frac{1}{n} \sum_{i=1}^{n} f(\theta, x_i)\}, \tag{1}$$

where $x_i \in \mathcal{D}$ is the i-th training sample of the total training dataset $\mathcal{D}$ and n is the number of training samples.

Stochastic Gradient Descent (SGD) is an optimization algorithm commonly used in LLM fine-tuning. It's a variant of the gradient descent algorithm that's designed to handle large datasets more efficiently.

Definition 2 (Stochastic gradient descent) . *SGD is a differentially private optimizer with learning rate η that updates parameters as*

$$\theta_t = \theta_{t-1} - \eta \cdot \nabla f(\theta_{t-1}, \xi_t), \tag{2}$$

where $\xi_t \in \mathcal{D}$ is the minibatch at time t and $\nabla f(\theta_{t-1}, \xi_t)$ is the average of gradients estimated by back propagation on ξ_t.

3.2 Differential Privacy

Differential privacy is a concept in data privacy that aims to provide a mathematical definition for the privacy guarantees of an algorithm or system.

Definition 3 (Differential Privacy [13]) . *A randomized mechanism $\mathcal{A} : \mathcal{X} \to \mathcal{S}$ is called (ε, δ) - differential private with respect to d if for every pair of adjacent datasets $X, X' \in \mathcal{X}$ satisfying $\mathrm{d}(X, X') \leq 1$ and every subset of outputs $s \in \mathcal{S}$ it holds that:*

$$\mathbb{P}[\mathcal{A}(X) \in s] \leq e^{\varepsilon} * \mathbb{P}\left[\mathcal{A}(X') \in s\right] + \delta, \tag{3}$$

where $\mathrm{d} : \mathcal{X}^2 \to [0, \infty)$ be the distance between two datasets.

Differentially Private Stochastic Gradient Descent (DPSGD) is a privacy-preserving variant of SGD. It is designed to train machine learning models while providing strong guarantees of differential privacy.

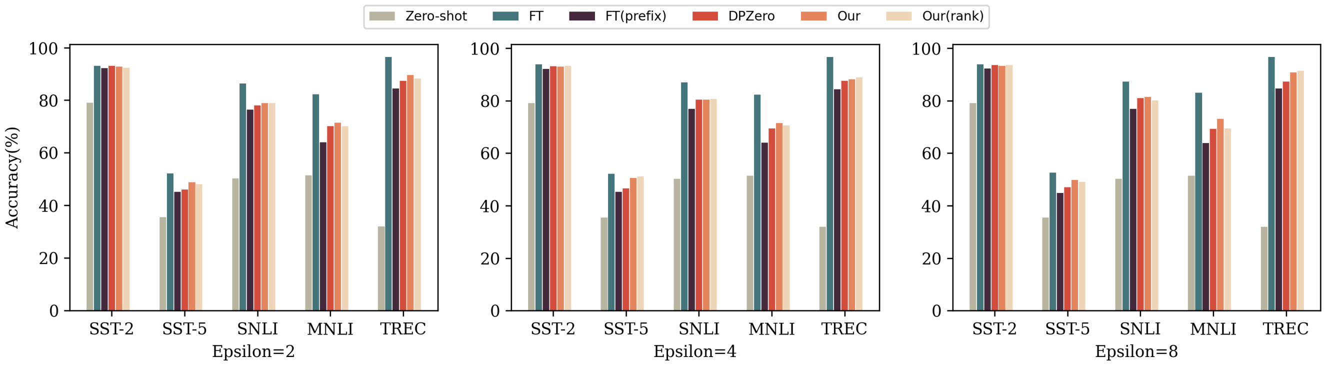

Figure 1: Experiments on RoBERTAa-large. We report zero-shot, DPZero [49], our method, our method with rank-based important matrix and fine-tuning (FT) with full parameter and prefix-tuning. Our method outperforms zero-shot, FT (prefix) and DPZero and approaches FT with much less memory (Detailed numbers in Table 3).

Definition 4 (DP-SGD [1]) . *DP-SGD is a differentially private optimizer with clip-threshold C, noise scale σ and learning rate η that updates parameters as*

$$\theta_t = \theta_{t-1} - \eta \cdot \frac{1}{m}\left(\sum_i \hat{g}(\theta_{t-1}, x_i) + N(0, \sigma^2 C^2 \boldsymbol{I}_d)\right), \qquad (4)$$

where $\hat{g}(\theta_{t-1}, x_i)$ is the gradient clipped by clip-threshold C, $\hat{g}(\theta_{t-1}, x_i) = g(\theta_{t-1}, x_i)/\max(1, \frac{||g(\theta_{t-1}, x_i)||_2}{C})$ and $g(\theta_{t-1}, x_i)$ denote the true gradient on data point x_i.

The following three lemmas are frequently used in the privacy analysis of DPSGD.

Lemma 1 (Privacy amplification by subsampling [20]) . *Let $\mathcal{A} : \mathcal{X}^{r_2} \to \mathcal{S}$ be a (ε, δ)-DP machanism. Then, the mechanism $\mathcal{A}' : \mathcal{X}^{r_1} \to \mathcal{S}$ defined by $\mathcal{A}' = \mathcal{A} \circ \mathrm{samp}_{r_1, r_2}$ is (ε', δ')-DP, where*

$$\varepsilon' = \log\left(1 + q\left(e^{\varepsilon} - 1\right)\right), \delta' = q\delta, q = \frac{r_2}{r_1}.. \qquad (5)$$

In particular, when $\varepsilon < 1, \mathcal{A}'$ is $(\mathcal{O}(q\varepsilon), q\delta)$-DP.

Lemma 2 (Strong composition [14]) *Let $\mathcal{A}_1, \ldots, \mathcal{A}_T$ be T adaptive (ε, δ)-DP mechanisms, where $\varepsilon, \delta \geq 0$. Then, for any $\delta' \geq 0$, the composed mechanism $\mathcal{A} = (\mathcal{A}_1, ..., \mathcal{A}_T)$ is $(\hat{\varepsilon}, \hat{\delta})$-DP, where*

$$\hat{\varepsilon} = \sqrt{2T \log\left(\frac{1}{\delta'}\right)}\varepsilon + T\varepsilon\left(e^{\varepsilon} - 1\right), \hat{\delta} = T\delta + \delta'. \qquad (6)$$

Moments accountant method tracks the accumulation of privacy loss over multiple iterations. By using the moments accountant, it is possible to estimate the overall privacy loss over a sequence of operations, which provides a tighter privacy analysis compared with strong composition Lemma 2.

Lemma 3 (Moments Accountant [1]) . *There exisi constant c_1 and c_2 so that given the sampling probability $q = m/n$ and the number of steps T, for any $\varepsilon < c_1 q^2 T$, DP-SGD is (ε, δ)-differentially private for any $\delta > 0$ if we choose*

$$\sigma \geq c_2 \frac{q\sqrt{T \log(1\delta)}}{\varepsilon}. \qquad (7)$$

3.3 Zeroth-order optimization

Zeroth-order gradient estimation does not rely on explicit gradient information to update the parameters of a model. In zeroth-order methods, function evaluations at different points in the parameter space are used to approximate the gradient.

Definition 5 (Zeroth-order gradient estimation [35]) . *Given a model with parameters $\theta \in \mathcal{R}^d$ and a loss function f, SPSA estimates the gradient on a minibatch ξ as*

$$\nabla f_\beta(\theta, \xi) = \frac{1}{2\beta}\left(f(\theta + \beta\mathbf{v}, \xi) - f(\theta - \beta\mathbf{v}, \xi)\right)\mathbf{v}. \qquad (8)$$

where $\mathbf{v} \in \mathcal{R}^d$ with $\mathbf{v} \sim N(0, \boldsymbol{I}_d)$ and β is the ZO scale parameter.

Gaussian Smoothing: Gaussian smoothing is a well-known technique that converts a possibly non-smooth function to a smooth approximation [28]. Given a differentiable function $f : \mathcal{R}^d \to \mathcal{R}$ and $\beta \leq 0$, we define the Gaussian smoothing of f as $f_\beta(\theta) = \mathbb{E}_{\mathbf{v} \sim \mathbb{V}}\left[f(\theta + \beta\mathbf{v})\right]$ where $\mathbb{V}$ is a standard Gaussian distribution. Gaussian smoothing has the following properties.

Lemma 4 *Suppose $f : \mathcal{R}^d \sim \mathcal{R}$ is differentiable and L-Lipschitz. Then* (i) *f_β is L-Lipschitz;* (ii)$||f_\beta(\theta) - f(\theta)|| \leq L\beta\sqrt{d}$; (iii)*$f_\beta$ is differentiable and $\frac{\sqrt{d}L}{\beta}$-smooth;* (iv)

$$\nabla f_\beta(\theta) = \mathbb{E}_{\mathbf{v} \sim \mathbb{V}}\left[\frac{1}{2\beta}\left(f(\theta + \beta\mathbf{v}) - f(\theta - \beta\mathbf{v})\right)\mathbf{v}\right]. \qquad (9)$$

Assumption 1 *The function $f_\beta(\theta, x)$ is L-Lipschitz and $\frac{\sqrt{d}L}{\beta}$-smooth for every x. There exist γ that for every $x \in \mathcal{D}$, we have*

$$\mathbb{E}\left[\left\|\nabla F_\beta(\theta) - \nabla f_\beta(\theta, x)\right\|^2\right] \leq \gamma^2. \qquad (10)$$

Assumption 2 *The function $f_\beta(\theta,\mathcal{D})$ satisfies PL condition if for any $\theta \in \mathcal{R}^d$ we have*

$$\|\theta-\theta^*\|^2 \le \frac{1}{2\mu}\left(f_\beta(\theta,\mathcal{D})-f_\beta(\theta^*,\mathcal{D})\right). \tag{11}$$

4 Methodology

In section 4.1, we introduce Differential Private Zeroth-order fine-tuning and describe our novel DP noise injection method for zeroth-order. In section 4.2, we propose Zeroth-order dradient estimation with dynamic ZO scale parameter. In section 4.3, we propose the first stagewise DP-ZO fine-tuning method to our knowledge and provide theoretical analysis for both privacy and convergence. In section 4.4, we design stagewise DP-ZO fine-tuning with data-free pruning to guide the model to update on important directions and further propose important matrix to improve model performance.

4.1 Differential Private Zeroth-Order Fine-tuning

The zeroth-order gradient is estimated by calculating the difference of the function on two points $\theta+\beta\mathbf{v}$ and $\theta-\beta\mathbf{v}$. Instead of clipping both the loss function $f(\theta+\beta\mathbf{v})$ and $f(\theta-\beta\mathbf{v})$, we clip the difference of the function on two points since the absolute value of the difference is much smaller than the loss function itself (3.2e-5 (0.007%) compared to 0.45 in average).

For privacy guarantee, Gaussian noise is injected into the clipped gradient during training. However, in the zeroth-order gradient descent algorithm, the direction of the approximate gradient $\nabla f_\beta(\theta)$, denoted by $\mathbf{v}$, is sampled from the standard Gaussian distribution independent of the private data.

Compared to adding Gaussian to all dimensions of the gradient in DP-SGD, we propose a novel method of noise injection by adding noise $N(0,\sigma^2C^2)$ to $\frac{1}{2\beta}\left(f(\theta+\beta\mathbf{v})-f(\theta-\beta\mathbf{v})\right)$, achieving (ε,δ)-DP guarantee. By standard post processing, the approximate gradient $\nabla f_\beta(\theta)$ is also (ε,δ)-DP since $\mathbf{v}$ is independent of private data.

4.2 Zeroth-order Gradient Estimation with Dynamic ZO scale parameter

The zeroth-order method suffers from utility loss owing to the bias between the actual gradient and the zeroth-order gradient. Zeroth order gradient is estimated by calculating the difference of the function on two points $\theta+\beta\mathbf{v}$ and $\theta-\beta\mathbf{v}$ where β is the ZO scale parameter. The choice of ZO scale parameter β has only been seen as a hyperparameter in [27] [17]. Shi et al. [33] pointed out that the ZO scale parameter should be as small as possible but can be set too small in practice since the accuracy of the computing system is limited. Yi et al. [45] proposed zeroth-order algorithms with a time-varying (decreasing) ZO scale parameter. However, when the model is close to convergence, the difference of the loss function on two points can be quite small such that the two-point estimated gradient can be quite large when the ZO scale parameter is small, contrary to the nature of the model approaching convergence. Thus, we propose dynamic zeroth-order gradient estimation with an increasing ZO scale parameter, reducing the bias of the zeroth-order gradient. More results are detailed in Table 6.

Table 1: Clipping bias with different ZO scale parameters. $\mathbb{P}$(Clip) denotes the possibility of getting clipped.

β	1e-6	2e-6	4e-6	6e-6
$\mathbb{P}$(Clip)	13.40%	12.20%	11.70%	11.60%

Furthermore, we find that zeroth-order gradients with bigger ZO scale parameters have. The lower the possibility of zeroth-order gradient getting clipped (detailed in Table 1).

4.3 Stagewise DP ZO fine tuning

First, let us present the algorithm that we intend to analyze in Algorithm 1. At the s-th stage, a regularized function $\phi_s(\theta)$ is constructed that consists of the original objective $f_{\beta_s}(\theta)$ and a quadratic regularizer $\frac{1}{2\lambda}\left\|\theta-\theta^{s-1}\right\|_2^2$. The reference point θ^{s-1} is a returned solution from the previous stage, which is also used for an initial solution for the current stage. Adding the strongly convex regularizer at each stage helps convert the weakly-convex loss function $f_{\beta_s}(\theta)$ to convex or strongly-convex $\phi_s(\theta)$. For each regularized problem, the SGD with a constant stepsize and ZO scale parameter is employed for a number of iterations with an appropriate returned solution. The procedure is described in Algorithm 2.

```
Algorithm 1: Stagewise DP Zeroth order optimizer($\theta^0,\lambda,\beta_0,S,T_0,\eta_0,\mathbb{V},\mathcal{D}$)
input : Initial point $\theta^0 \in \mathcal{R}^d$, initial stepsize $\eta_0 \ge 0$,
        initial ZO scale parameter $\beta_0$, regularization
        parameter $\lambda$, initial iteration number $T_0$,
        number of epoch $S$
for $s = 1$ to $S$ do
    $\beta_s = k\cdot\beta_{s-1},\ T_s = 2\cdot T_{s-1},\ \eta_s = \eta_{s-1}/2$;
    $\phi_s(\theta) = f_{\beta_s}(\theta) + \frac{1}{2\lambda}\left\|\theta-\theta^{s-1}\right\|_2^2$;
    $\theta^s = $ DP-ZOO$(\phi_s(\theta),\theta^{s-1},\beta_s,T_s,\eta_s,\mathbb{V},\mathcal{D})$
end
return final model parameter $\theta^S$
```

Algorithm 2 is a **D**ifferential **P**rivate **Z**eroth-**O**rder **O**ptimizer that update parameters in specific distribution $\mathbb{V}$. In the step t of Algorithm 2, a random set of ξ_t is sampled from dataset $\mathcal{D}$. Random P vectors $\{\mathbf{v}_t^p\}_{p=1}^P \in \mathbb{V}$ are sampled from distribution $\mathbb{V}$. For vector $\mathbf{v}_t^p$, we estimate the gradient step $\mathrm{gra}_{f,t}^{i,p}$ of original objective $f(\theta_{t-1})$ on every data point x_t^i in line 7 and then clip it to $\hat{\mathrm{gra}}_{f,t}^{i,p}$ in L2-norm. The gradient step of the quadratic regularizer is added to the clipped gradient step as $\mathrm{gra}_t^{i,p}$ in line 9. Gaussian noise is added to the sum of $\mathrm{gra}_t^{i,p}$ and then average. Timing the direction $\mathbf{v}_t^p$ is the gradient $g_p(\theta_{t-1})$ on direction $\mathbf{v}_t^p$. The gradient in iteration t is $g(\theta_{t-1})$ by taking average over P directions. Parameters are then updated by $g(\theta_{t-1})$ with learning rate η_s.

```
Algorithm 2: DP-ZOO($\phi_s(\theta), \theta^{s-1}, \beta_s, T_s, \eta_s, \mathbb{V}, \mathcal{D}$)
   input : stepsize $\eta_s \geq 0$, ZO scale parameter $\beta_s$,
           parameter dimension $d$, sample dataset $\xi_t$,
           sample size $m$, clipping threshold $C$, vector
           distribution $\mathbb{V}$ and iteration number $T_s$
1  Set initial parameter $\theta_0 = \theta^{s-1}$;
2  for $t = 1$ to $T_s$ do
3      Randomly sample $\xi_t = \{x_t^i\}_{i=1}^m$ from dataset $\mathcal{D}$;
4      Sample $P$ random vectors $\{\mathbf{v}_t^p\}_{p=1}^P \in \mathbb{V}$;
5      for $p = 1$ to $P$ do
6          for $i = 1$ to $m$ do
7              $\mathrm{gra}_{f,t}^{i,p} =$
               $\frac{1}{2\beta_s}\left(f(\theta_{t-1} + \beta_s \mathbf{v}_t^p, x_t^i) - f(\theta_{t-1} - \beta_s \mathbf{v}_t^p, x_t^i)\right)$ ;
8              Clip : $\hat{\mathrm{gra}}_{f,t}^{i,p} \leftarrow \mathrm{gra}_{f,t}^{i,p} / \max\left\{1, \frac{|\mathrm{gra}_{f,t}^{i,p}|}{C}\right\}$;
9              $\mathrm{gra}_t^{i,p} = \hat{\mathrm{gra}}_{f,t}^{i,p} + \frac{1}{\lambda}\|\theta_{t-1} - \theta_0\|$;
10         end
11         $g_p(\theta_{t-1}) = \frac{1}{m}\left(\sum_{i=1}^m \mathrm{gra}_t^{i,p} + N(0, \sigma^2 C^2)\right) \cdot \mathbf{v}_t^p$;
12     end
13     $g(\theta_{t-1}) = \frac{1}{P}\sum_{p=1}^P g_p(\theta_{t-1})$;
14     $\theta_t = \theta_{t-1} - \eta_s \cdot g(\theta_{t-1})$;
15 end
16 return $\theta_{T_s}$
```

Theorem 1 (Privacy analysis of Algorithm 1) *In every stage of Algorithm 1 and in iteration t of DP-ZOO, a random set ξ_t of m samples are sampled out of dataset $\mathcal{D}$ and P random vectors are sampled from standard Gaussian distribution. Gaussian noise is added to the sum of zeroth-order on dataset ξ_t in direction $\mathbf{v}_t^p$ after clipping $\left|\mathrm{gra}_{f,t}^{i,p}\right|$ to C. There exist constants c_1 and c_2, for any $\varepsilon < c_1 m^2 T / n^2$, the overall algorithm is (ε, δ)-DP if*

$$\sigma^2 \geq \frac{c_2^2 P^2 m^2 T \log(P/\delta)}{\varepsilon^2 n^2}. \tag{12}$$

Proof 4.1 *In the algorithm, the sensitivity of the sum of ZO estimated $\sum_{i=1}^m \hat{\mathrm{gra}}_{f,t}^{i,p}$ is clipped to C. The zeroth-order gradient on data point x_t^i is estimated in P random directions such that the privacy budget is divided equally into P. Vectors are randomly sampled from standard Gaussian distribution, thus by post processing, the privacy of the zeroth-order gradient is guaranteed.*

Lemma 5 *Under Assumption 1 and $f(\theta, x)$ is a ρ-weakly-convex function of θ, by applying DP-ZOO (Algorithm 2) with $\eta_s \leq \frac{\beta_s}{\sqrt{d}L}$, for any $\theta \in \mathcal{R}^d$, we have*

$$\mathbb{E}[\phi_s(\theta_{T_s}) - \phi_s(\theta)] \leq \left(\frac{1}{2\eta_s T_s} + \frac{1}{2T_s\lambda}\right)\|\theta_0 - \theta\|^2 + \eta_s\left(\frac{dc_2^2 C^2 PT \log(P/\delta)}{\varepsilon^2 n^2} + \frac{64d\beta_s^2 L^4}{Pm} + \frac{\gamma^2}{Pm} + \frac{8dC^2}{ePm}\right). \tag{13}$$

Theorem 2 *Suppose Assumption 1 holds and $f(\theta, x)$ is ρ-weakly-convex of θ. Then by setting $\lambda = \frac{3}{2\mu}$, $\eta_s T_s = \frac{3}{2\mu}$ and $\eta_s = \alpha_{s-1} \cdot \min\left\{\frac{Pm}{6\gamma^2}, \frac{Pme}{48dC^2}, \frac{\varepsilon^2 n^2}{6dc_2^2 C^2 PT \log(P/\delta)}, \frac{Pm}{384d\beta_s^2 L^4}\right\}$, after $S = \lceil \log(\alpha_0/\alpha) \rceil$ stages, we have*

$$\phi_s(\theta^S) - \phi_s(\theta^*) \leq \alpha. \tag{14}$$

The total ZO oracle complexity is

$$\mathcal{O}\left(\left(\gamma^2 + dC^2 + d\beta_S^2 L^4 + \frac{dC^2 P^2 m \log(P/\delta)}{\varepsilon^2 n^2}\right) \cdot \frac{1}{\mu\alpha}\right). \tag{15}$$

4.4 Pruning with Important Matrix

In differentially private zeroth-order method, pruning helps the model to update on more important directions. Especially in private settings, data-free pruning is essential to enhance zeroth-order optimization whose pseudo-code is described in Algorithm 3.

In Algorithm 3, a new loss function $\mathcal{L}(\theta)$ is defined, where $\mathbb{1}$ is the all ones vector and $|\theta^{[l]}|$ is the element-wise absolute value of parameters in the l-th layer. Synaptic saliency scores $\nabla \mathcal{L}_p \odot \theta$ are esitimated. The scores will be used in Algorithm 4 to update the vector distribution which helps with convergence.

In DP fine-tuning, data-free pruning will not cause extra privacy concerns since no private data is used. Pruning freezes most of the pre-trained parameters and modifies the pre-trained model with small trainable parameters.

Pruning can be seen as radically setting the direction of the gradient for frozen parameters to be sampled from $N(0,0)$. We further propose **Importance Matrix** to guide the training of the remaining parameters. Parameters with high scores are sampled from $N(0,x)(x > 1)$ with larger weight, tuned with more effort. The experiment result is shown in Table 3. We propose rank-based important matrix whose standard deviation follows the uniform distribution from A to B based on the rank of parameters remained.

```
Algorithm 3: Data-free ZO pruning($\theta, r, type$)
input :Model parameter $\theta$, pruning rate $r$ and matrix
       property $type$
Define $\mathcal{L}(\theta) \leftarrow \mathbb{1}^\top \left(\prod_{l=1}^{L} |\theta^{[l]}|\right) \mathbb{1}$;
Initialize binary mask: $mask = \mathbb{1}$;
Sample $P$ random vectors $\{\mathbf{v}_p\}_{p=1}^{P} \in N(0, \mathbf{I}_d)$ ;
for $p = 1$ to $P$ do
    $\nabla \mathcal{L}_p = \frac{1}{2\beta} \left(\mathcal{L}(\theta + \beta \mathbf{v}_p) - \mathcal{L}(\theta - \beta \mathbf{v}_p)\right) \mathbf{v}_p$
end
$Score \leftarrow \frac{1}{P} \sum_{p=1}^{P} \nabla \mathcal{L}_p \odot \theta$;
Update vector distribution $\mathbb{V}$ ;
$\mathbb{V} \leftarrow$ Importance Matrix$(Score, r, type) \cdot N(0, \mathbf{I}_d)$
```

```
Algorithm 4: Importance Matrix($Score, r, type$)
input :Pruning rate $r$, Matrix type $type$, list $Score$,
       upper bound $A$,lower bound $B$
Initial importance matrix $M = 0_{[d \times d]}$;
$Score = Score$.sort();
$Score = Score[r \cdot \mathrm{len}(Score) :]$;
for $i$-th dimension $\theta_i$ of parameter $\theta$ do
    if $\theta_i.score \in Score$ then
        if $type == pruning - only$ then
            $M[i][i] = 1$
        end
        if $type == rank - based$ then
            $rank = Score.\mathrm{index}(\theta_i.score)$;
            $M[i][i] = A - \frac{(A-B) \cdot rank}{r \cdot \mathrm{len}(Score)}$
        end
    end
end
return importance matrix $M$
```

4.5 Stagewise DP ZO Fine-Tuning with pruning

We present the procedure code of stagewise DP-zo fine tuning with pruning in Algorithm 5. Algorithm 5 can be divided into two phases. First, we employ data-free pruning to find the important parameters of θ (the parameters to be fine-tuned). Next, we use ZO-based fine-tuning method to optimize the model on $\mathbb{V}$. We denote $\theta' \in \mathcal{R}^{r \cdot d}$ as the parameters to be tuned and $h(\theta')$ as the objective function to be fine-tuned. We use $\psi_s(\theta') = h(\theta') + \frac{1}{2\lambda} ||\theta' - \theta'^{s-1}||_2^2$ to denote the regularized function.

Theorem 3 (Privacy analysis of Algorithm 5) *Since pruning in the first phase of Algorithm 5 does not require private data, the calculated vector distribution $\mathbb{V}$ will not leak any information about private data. Thus, Algorithm 5 shares the same privacy guarantee with Algorithm 1.*

```
Algorithm 5: Stagewise DP ZO with pruning
input :Model parameter $\theta$, stepsize $eta$, ZO scale
       parameter $\beta$, iteration numbers $T$, number of
       epoch $S$, pruning rate $r$, dataset $\mathcal{D}$ and matrix
       property $type$
$\mathbb{V} \leftarrow$ Algorithm3$(\theta, r, type)$;
                              ▷ To find important parameters;
$\theta_{\mathrm{out}} \leftarrow$ Algorithm1$(\theta, \lambda, \beta, S, T, \eta, \mathbb{V}, \mathcal{D})$;
                    ▷ Fine-tune model on $\mathbb{V}$ with ZO method;
return final model parameters $\theta_{\mathrm{out}}$
```

Theorem 4 *Suppose Assumption 1 holds on $\theta' \in \mathcal{R}^{r \cdot d}$ and loss function $h(\theta', x)$ is ρ-weakly-convex of θ'. Then by setting $\lambda = \frac{3}{2\mu}$, $\eta_s T_s = \frac{3}{2\mu}$ and $\eta_s = \alpha_{s-1} \cdot \min\left\{\frac{Pm}{6\gamma^2}, \frac{Pme}{48dC^2}, \frac{\epsilon^2 n^2}{6rdc_2^2 C^2 PT \log(P/\delta)}, \frac{Pm}{384rd\beta_s^2 L^4}\right\}$, after $S = \lceil \log(\alpha_0/\alpha) \rceil$ stages, we have*

$$\psi_s(\theta'^S) - \psi_s(\theta^*) \leq \alpha. \tag{16}$$

The total ZO oracle complexity of Algorithm 5 is

$$\mathcal{O}\left(\left(\gamma^2 + rdC^2 + rd\beta_S^2 L^4 + \frac{rdC^2 P^2 m \log(P/\delta)}{\epsilon^2 n^2}\right) \cdot \frac{1}{\mu\alpha}\right). \tag{17}$$

Algorithm 5 greatly reduces the total complexity by a factor of $1/r$ compared with Algorithm 1.

5 Experiment

We conduct comprehensive experiments on both medium-sized masked LM (RoBERTa-large, 350M [26]) and large autoregressive LM (OPT-2.7B [52]) in few-shot settings. We also explore both full-parameter tuning and parameter-efficient fine-tuning like prefix-tuning.

5.1 Experimental Setup

We report the metric of accuracy on downstream tasks that runs with random seed=42. We detail the full hyperparameter settings of our experiments in Appendix A.1.1.

Datasets: For RoBERTa-large, we consider classification datasets: SST-2 [34], SST-5 [34], SNLI [2], MNLI [44] and TREC [40]. We randomly samples 1000 examples for testing amd have 512 examples per class for both training and validation. For OPT experiments, we consider the SuperGLUE dataset collection [42] including: CB [10], BoolQ [9], MultiRC [21] and ReCoRD [51]. We also include SST-2 [34] in our experiments. We randomly sample 1024 examples for training, 500 examples for validation and 1000 examples for testing.

Models We conduct experiments on both masked language

Table 2: Experiments on RoBERTAa-large. With pruning denotes the best performance of Algorithm 5 among pruning rate $r = \{0.5\%, 1\%, 2\%, 5\%, 10\%\}$. Without pruning represents Algorithm 1.

Task	**SST-2**	**SST-5**	**SNLI**	**MNLI**	**TREC**
Small Privacy budget: $\varepsilon = 2$					
Without pruning	92.9	45.8	76.8	66.3	83.4
With pruning	92.7 (0.2 ↓)	48.8 (3.0 ↑)	78.9 (2.1 ↑)	71.4 (5.1 ↑)	89.6 (6.2 ↑)
Medium Privacy budget: $\varepsilon = 4$					
Without pruning	92.6	42.0	78.9	67.3	83.8
With pruning	93.0 (0.4 ↑)	50.5 (8.5 ↑)	80.4 (1.5 ↑)	71.5 (4.2 ↑)	88.2 (4.4 ↑)
Large Privacy budget: $\varepsilon = 8$					
Without pruning	93.2	41.2	76.5	69.2	84.2
With pruning	93.2 (—)	49.8 (8.6 ↑)	81.4 (4.9 ↑)	73.1 (3.9 ↑)	90.8 (6.6 ↑)

.

model (RoBERTa-large, 350M [26]), autoregressive language model (OPT-2.7B [52]), and LLaMA-7B (to be completed) and GPT-neo-1.3B (to be completed) in few-shot settings. We present our main results in Table 3,4. We include a range of ablation studies, including varying the dataset size and the pruning rate. We also explore both full-parameter tuning and parameter-efficient fine-tuning like prefix-tuning. We present our main results in

Privacy Budgets: We consider various privacy levels with $\varepsilon = [2, 4, 8]$ and dynamic $\delta = 1/n$ where n is the number of private data for (ε, δ)-DP. We also include $\varepsilon = \infty$ baseline that is trained without adding DP noise. In our experiments, we set the clipping threshold to $C = 30$. We include the zero-shot does not incur any privacy loss because we evaluate the pre-trained model directly without finetuning on private data.

5.2 Main Results

We conduct experiments with RoBERTa-large on sentiment classification, natural language inference and topic classification. We follow [27] [16] to study the few-shot and many-shot settings, sampling k examples per class for $k = 512$. DPZero [49] is tested with moment accountant just for fairness. We summarize the results from Table 3 below.

Our method works significantly better than zero-shot and other memory-equivalent methods. On all five diverse tasks, our method can optimize the pre-trained model and consistently perform better than zero-shot and prefix-tuning. We also show for several tasks that our method can outperform DPZero up to 5% even with the same privacy analysis. Our method outperforms 1.2% on SST-2 with $\varepsilon = 4$ compared with the results shown in Table 1 of [38].

Pruning improves model performance in private settings. We compare Algorithm 5 (with pruning) and Algorithm 1 (without pruning). We show the best performance of Algorithm 5 among five different pruning rates in Table 2. Pruning improves performance greatly in almost five tasks in private settings ($\varepsilon = 2, 4, 8$). As a detailed result, the best performance of Algorithm 5 is 71.5% on MNLI, outperforming Algorithm 20 (without pruning) by 4.2%. In different private settings, the optimal pruning rate varies which will be discussed in Section 5.5.

Table 3: Experiments on RoBERTAa-large. Our method outperforms zero-shot, DPZero and approaches FT with much less memory.

Task	**SST-2**	**SST-5**	**SNLI**	**MNLI**	**TREC**
Type	—sentiment—		NL inference		topic
Zero-shot	79.0	35.5	50.2	51.4	32.0
Small Privacy budget: $\varepsilon = 2$					
FT	93.0	52.1	86.4	82.2	96.4
FT-prefix	92.2	45.2	76.4	64.0	84.4
DPZero	**93.1**	46.1	78.0	70.1	87.4
Ours	92.7	**48.8**	**78.9**	**71.4**	**89.6**
Ours (rank)	92.4	48.3	**78.9**	70.2	89.0
Medium Privacy budget: $\varepsilon = 4$					
FT	93.8	52.1	87.0	82.3	96.6
FT-prefix	92.1	45.2	76.8	64.0	84.4
DPZero	93.1	46.6	80.4	69.4	87.6
Ours	93.0	50.5	80.4	**71.5**	88.2
Ours (rank)	**93.3**	**51.1**	**80.7**	71.1	**90.4**
Large Privacy budget: $\varepsilon = 8$					
FT	93.8	52.6	87.2	83.0	96.6
FT-prefix	92.2	44.8	76.9	63.9	84.6
DPZero	93.4	47.0	80.9	69.2	87.2
Ours	93.2	**49.8**	**81.4**	**73.1**	90.8
Ours (rank)	**93.5**	49.6	80.7	70.8	**91.4**

With the promising results from RoBERTa-large, we extend our method to the OPT-2.7B [52]. We select SuperGLUE tasks [42] (including classification and multiple-choice). We randomly sample 1024 examples for training and 1000 test examples for each dataset.

Table 4: Experiments on OPT-2.7B with privacy budget $\varepsilon = 4, \delta = 1/1024$. ICL: in-context learning; FT-prefix for prefix-tuning. Our method has better performance than zero-shot, ICL and DPZero with almost the same memory consumption.

Task	**SST-2**	**CB**	**BoolQ**	**MultiRC**	**ReCoRD**
Zero-shot	56.3	50.0	48.0	44.3	75.8
ICL	77.6	62.5	57.9	47.8	75.7
DPZero	91.5	69.6	63.6	61.9	76.0
Ours	93.2	69.7	63.7	61.9	76.3

Our method works significantly better than zero-shot and ICL. On both classification and multiple-choice tasks, our method exhibits strong performance. On a 2.7B-parameter scale, our method outperforms zero-shot and ICL across all tasks with equivalent memory consumption (details in Table 4).

Our method outperforms DPZero [49] in all five tasks on autoregressive language OPT-2.7B, but not as significantly as in RoBERTa-large which we blame for the poor quality of data-free pruning and fixed pruning mask which can be updated during training. How to enhance the performance of pruning is worth further studying and We leave it as future work.

5.3 Memory usage

In this section, we profile the memory usage of zero-shot, ICL, FT, FT-prefix and our method. We test OPT-1.3B and OPT-2.7B with Nvidia A100 GPUs (40GB memory) on SST-2 and report the GPU memory consumption in Table 5.

Our method uses pruning mask to freeze model parameters during training which costs quite large GPU memory. However, pruning rate is usually set quite small which means the pruning mask is a sparse vector. We store non-zero ids of the mask instead of the whole vector to reduce memory consumption overhead. Pruning costs a slight amount of (8%) GPU memory while improving the model utility greatly.

Table 5: GPU memory comsumption with OPT-1.3B, OPT-2.7B and tuning on SST-2. Our method takes $r = 1\%$ as pruning rate.

GPU memory(MB)	OPT-1.3B	OPT-2.7B
Zero-shot	2517.73	5066.17
ICL	2517.73	5066.17
Ours	2737.03 (1.08×)	5514.65 (1.08×)
FT-prefix	2528.67	5080.55
FT	7545.08 (3×)	20250.93 (4×)

As shown in Figure 5, our method exhibits almost the same (1.08×) memory consumption which offers memory savings of up to 4 times compared to standard FT.

5.4 Dynamic ZO scale parameter

We fine-tune RoBERTa-large on SST-2 with both fixed and dynamic ZO scale parameter to demonstrate the superiority of dynamic ZO scale parameter.

Table 6: Training loss of RoBERTa-large on SST-2 with both fixed and dynamic ZO scale parameter. Dynamic ZO scale parameter is reduced during stages. We denote M1 as the scale parameter decreasing from $1e-5 \sim 1e-4$, M2 as $1e-6 \sim 1e-5$ and M3 as $1e-6 \sim 1e-4$.

Fix ZO scale parameter			Dynamic ZO scale parameter		
1e-4	1e-5	1e-6	M1	M2	M3
0.31532	0.31530	0.31639	0.31812	0.31434	0.31368

All experiments are done under $\varepsilon = 4$ with 6k steps. M3 represents the algorithm with ZO scale parameter increasing from $1e-6$ to $1e-4$ as the stage advances. It is shown in Table 6 that dynamic ZO scale parameter method M3 ($1e-6 \sim 1e-4$) exhibits the lowest loss on the training set compared with both fixed ZO scale parameter $1e-4$ and $1e-6$. Dynamic M2 and M3 both perform better than all fixed ZO scale parameter, further elaborating on the superiority of the dynamic ZO scale parameter approach.

5.5 Data-free Pruning

In the section, we first present our results of the utility of RoBERTa-large fine-tuned on downstream task SNLI with different learning rates and pruning rates in Table 11, aiming to find appropriate learning rates for different pruning rates. We conduct our following experiments under appropriate learning rates presented in Table 11. We further study the trends in optimal pruning rates selection under different privacy settings.

Pruning works well in private ZO fine-tuning. We show our results in Figure 2 that under all private settings, zeroth order fine-tuning with pruning achieves better performance (above 2%, more detailed numbers in Table 7). Especially when privacy budget is large like $\varepsilon = 8, \infty$, zeroth-order fine-tuning with pruning outperforms without pruning by $4 \sim 5\%$

Optimal pruning rate changes with privacy budget. In our experiments, pruning rate is seen a hyperparameter. However, the choice of pruning rate affect the model utility on downstream tasks (see Figure 2). When privacy budget is large (like $\varepsilon = 8$), useful parameters can be tuned well, small pruning rate can be chosen to lower the scale of DP noise discussed in section 4.4. When privacy budget is small (like $\varepsilon = 2$), 0.5% parameters can not be tuned well for large DP noise, thus bigger pruning rate $r = 10\%$ should be chosen for better convergence.

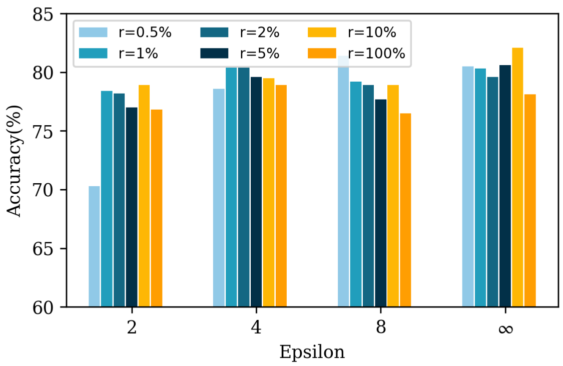

Figure 2: Fine tuning RoBERTa-large model on dataset SNLI with different pruning rates. Pruning rate r denotes the ratio of parameters to be tuned. Optimal r scales up with the decrease of privacy budget. Detailed numbers in Table 7.

Table 7: Experiments of RoBERTa-large fine-tuned on SNLI with different pruning rates

Pruning Rate	$\varepsilon = 2$	$\varepsilon = 4$	$\varepsilon = 8$	$\varepsilon = \infty$
$r = 0.5\%$	70.3	78.6	**81.4**	80.5
$r = 1\%$	78.4	**80.4**	79.2	80.3
$r = 2\%$	78.2	**80.4**	78.9	79.6
$r = 5\%$	77.0	79.6	77.7	80.6
$r = 10\%$	**78.9**	79.5	78.9	**82.1**
$r = 100\%$	76.8	78.9	76.5	78.1

5.6 Important Matrix

We further conduct experiments on pruning-only method and rank-based important matrix with different settings of intervals (upper bound and lower bound in Algorithm 4). It is shown in Table 8 that rank-based important matrix outperforms pruning-only method under medium and large privacy budgets while behaving poorly under small privacy budgets.

We point out that the optimal interval varies with privacy budget ε. Taking the interval [0.8-1.2] as an example, the step of zeroth-order gradient on the most important parameter will be sampled from $N(0, 1.2)$, thus the faster to converge but at the same time the noise introduced is also larger than other parameters in expectation which makes the parameter hard to find its saddle point. Things get extremely worse with small privacy budget. When $\varepsilon = 2$, parameters can not be tuned well in the first place, fine-tuning with important matrix makes it even worse. Thus, the optimal interval of important matrix under different private settings remains an open question to discuss.

Table 8: Rank-based important matrix with different intervals on RoBERTa-large.

Interval	[0.7-1.3]	[0.8-1.2]	[0.9-1.1]	[1.0-1.0]
Small Privacy budget: $\varepsilon = 2$				
SST-2	91.7	92.1	92.4	**92.7**
SST-5	47.3	48.3	48.1	**48.8**
SNLI	78.6	78.8	**78.9**	**78.9**
MNLI	68.7	70.2	70.1	**71.4**
TREC	88.4	89.0	88.2	**89.6**
Medium Privacy budget: $\varepsilon = 4$				
SST-2	92.2	**93.3**	93.2	93.0
SST-5	49.2	49.2	**51.1**	50.5
SNLI	80.7	**80.4**	80.1	80.4
MNLI	69.2	71.1	70.6	**71.5**
TREC	90.2	**90.4**	88.8	88.2
Large Privacy budget: $\varepsilon = 8$				
SST-2	90.8	**93.5**	**93.5**	93.2
SST-5	49.6	48.5	49.1	**49.8**
SNLI	78.7	80.1	78.0	**81.4**
MNLI	70.8	70.2	69.4	**73.1**
TREC	89.2	89.4	**91.4**	90.8

6 Conclusion

We have provided both theoretical and empirical studies among all concurrent works. Detailed theoretical proof of both privacy and convergence for both our algorithm are provided. We have shown that stagewise DP Zeroth-order method with pruning is memory-efficient and can effectively optimize large LMs across many tasks and scales. Further experiments suggest that the optimal pruning rate varies with the privacy budget ε. As a limitation, we did not explore stagewise DP Zeroth-order method with adaptive pruning rate which we hope to investage in the furture.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[2] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015.

[3] Zhiqi Bu, Yu-Xiang Wang, Sheng Zha, and George Karypis. Automatic clipping: Differentially private

deep learning made easier and stronger. *arXiv preprint arXiv:2206.07136*, 2022.

[4] Mark Bun, Jonathan Ullman, and Salil Vadhan. Fingerprinting codes and the price of approximate differential privacy. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 1–10, 2014.

[5] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 267–284, 2019.

[6] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021.

[7] Aochuan Chen, Yimeng Zhang, Jinghan Jia, James Diffenderfer, Jiancheng Liu, Konstantinos Parasyris, Yihua Zhang, Zheng Zhang, Bhavya Kailkhura, and Sijia Liu. Deepzero: Scaling up zeroth-order optimization for deep model training. *arXiv preprint arXiv:2310.02025*, 2023.

[8] Zaiyi Chen, Zhuoning Yuan, Jinfeng Yi, Bowen Zhou, Enhong Chen, and Tianbao Yang. Universal stagewise learning for non-convex problems with convergence on averaged solutions. *arXiv preprint arXiv:1808.06296*, 2018.

[9] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*, 2019.

[10] Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23, pages 107–124, 2019.

[11] Haonan Duan, Adam Dziedzic, Nicolas Papernot, and Franziska Boenisch. Flocks of stochastic parrots: Differentially private prompt learning for large language models. *arXiv preprint arXiv:2305.15594*, 2023.

[12] Cynthia Dwork. Differential privacy. In *International colloquium on automata, languages, and programming*, pages 1–12. Springer, 2006.

[13] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[14] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[15] Jun Feng, Laurence T Yang, Bocheng Ren, Deqing Zou, Mianxiong Dong, and Shunli Zhang. Tensor recurrent neural network with differential privacy. *IEEE Transactions on Computers*, 2023.

[16] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*, 2020.

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

[18] Bin Gu, Xiyuan Wei, Shangqian Gao, Ziran Xiong, Cheng Deng, and Heng Huang. Black-box reductions for zeroth-order gradient algorithms to achieve lower query complexity. *The Journal of Machine Learning Research*, 22(1):7685–7731, 2021.

[19] Jiyan He, Xuechen Li, Da Yu, Huishuai Zhang, Janardhan Kulkarni, Yin Tat Lee, Arturs Backurs, Nenghai Yu, and Jiang Bian. Exploring the limits of differentially private deep learning with group-wise clipping. *arXiv preprint arXiv:2212.01539*, 2022.

[20] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011.

[21] Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 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)*, pages 252–262, 2018.

[22] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip HS Torr. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*, 2018.

[23] Xuechen Li, Florian Tramer, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. *arXiv preprint arXiv:2110.05679*, 2021.

[24] Yansong Li, Zhixing Tan, and Yang Liu. Privacy-preserving prompt tuning for large language model services. *arXiv preprint arXiv:2305.06212*, 2023.

[25] Haolin Liu, Chenyu Li, Bochao Liu, Pengju Wang, Shiming Ge, and Weiping Wang. Differentially private learning with grouped gradient clipping. In *ACM Multimedia Asia*, pages 1–7. 2021.

[26] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

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

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

[29] Ashwinee Panda, Tong Wu, Jiachen T Wang, and Prateek Mittal. Differentially private in-context learning. *arXiv preprint arXiv:2305.01639*, 2023.

[30] Spyridon Pougkakiotis and Dionysis Kalogerias. A zeroth-order proximal stochastic gradient method for weakly convex stochastic optimization. *SIAM Journal on Scientific Computing*, 45(5):A2679–A2702, 2023.

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

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

[33] Wanli Shi, Hongchang Gao, and Bin Gu. Gradient-free method for heavily constrained nonconvex optimization. In *International Conference on Machine Learning*, pages 19935–19955. PMLR, 2022.

[34] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

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

[36] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune bert for text classification? In *Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18*, pages 194–206. Springer, 2019.

[37] Hidenori Tanaka, Daniel Kunin, Daniel L Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. *Advances in neural information processing systems*, 33:6377–6389, 2020.

[38] Xinyu Tang, Ashwinee Panda, Milad Nasr, Saeed Mahloujifar, and Prateek Mittal. Private fine-tuning of large language models with zeroth-order optimization. *arXiv preprint arXiv:2401.04343*, 2024.

[39] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[40] 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*, pages 200–207, 2000.

[41] Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge university press, 2019.

[42] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32, 2019.

[43] Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. *arXiv preprint arXiv:2002.07376*, 2020.

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

[45] Xinlei Yi, Shengjun Zhang, Tao Yang, and Karl H Johansson. Zeroth-order algorithms for stochastic distributed nonconvex optimization. *Automatica*, 142:110353, 2022.

[46] Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, et al. Differentially private fine-tuning of language models. *arXiv preprint arXiv:2110.06500*, 2021.

[47] Da Yu, Huishuai Zhang, Wei Chen, Jian Yin, and Tie-Yan Liu. Large scale private learning via low-rank reparametrization. In *International Conference on Machine Learning*, pages 12208–12218. PMLR, 2021.

[48] Zhuoning Yuan, Yan Yan, Rong Jin, and Tianbao Yang. Stagewise training accelerates convergence of testing error over sgd. *Advances in Neural Information Processing Systems*, 32, 2019.

[49] Liang Zhang, Kiran Koshy Thekumparampil, Sewoong Oh, and Niao He. Dpzero: Dimension-independent and differentially private zeroth-order optimization. *International Workshop on Federated Learning in the Age of Foundation Models in Conjunction with NeurIPS 2023*, 2023.

[50] Renrui Zhang, Jiaming Han, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, Peng Gao, and Yu Qiao. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*, 2023.

[51] Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*, 2018.

[52] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

[53] Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A Appendix

A.1 Experiment setup

A.1.1 Hyperparameter

For RoBERTa-large, we conduct experiments under the hyperparameters detailed in Table 9. We use the hyperparameters in Table 10 for our experiments on OPT-2.7B.

For privacy concern, we do not use early stopping in our experiments. The final model is saved and tested on 1000 test examples (Table 3, 4)

Table 9: The hyperparameters grids used for RoBERTa-large experiments. All experiments use 6K steps for training.

Experiment	Hyperparameters	Values
DPZero	Batch size	64
	Learning rate	$\{2e-5, 1e-5, 5e-6\}$
	ZO scale parameter	1e-6
Our method	Batch size	64
	Learning rate	$\{2e-4, 1e-4, 4e-5, 2e-5, 1e-5\}$
	ZO scale parameter	1e-6 to 1e-5
	Pruning rate (r)	$\{0.5\%, 1\%, 2\%, 5\%, 10\%, 100\%\}$
	Stage size	3
	Regulation parameter	5e-4
DP-SGD	Batch size	64
	Learning rate	$\{1e-4, 5e-4, 1e-3, 5e-3\}$
DP-prefix	Batch size	64
	Learning rate	$\{1e-2, 3e-2, 5e-2\}$
	prefix tokens	5

Table 10: The hyperparameters grids used for OPT experiments. All experiments use 6K steps for training.

Experiment	Hyperparameters	Values
ICL	Examples	32
DPZero	Batch size	16
	Learning rate	$\{1e-6\}$
	ZO scale parameter	1e-4
Our method	Batch size	16
	Learning rate	$\{4e-5, 3e-5, 2e-5, 1e-5, 5e-6\}$
	ZO scale parameter	1e-4 to 4e-4
	Pruning rate (r)	$\{0.5\%, 1\%, 2\%, 5\%\}$
	Stage size	3
	Regulation parameter	5e-4

A.2 Complete Proof

Lemma 6 *Under Assumption 1 and $f(\theta, x)$ is a ρ-weakly-convex function of θ, by applying DP-ZOO (Algorithm 2) with $\eta_s \leq \frac{\beta_s}{\sqrt{d}L}$, for any $\theta \in \mathcal{R}^d$, we have*

$$\mathbb{E}[\phi_s(\theta_{T_s}) - \phi_s(\theta)] \leq \left(\frac{1}{2\eta_s T_s} + \frac{1}{2T_s\lambda}\right) \|\theta_0 - \theta\|^2 + \eta_s \left(\frac{dc_2^2 C^2 PT \log(P/\delta)}{\varepsilon^2 n^2} + \frac{64 d\beta_s^2 L^4}{Pm} + \frac{\gamma^2}{Pm} + \frac{8dC^2}{ePm}\right). \tag{18}$$

Proof for Lemma 5:

Recall that $\phi_s(\theta) = f_{\beta_s}(\theta) + \frac{1}{2\lambda}\|\theta_t - \theta^s\|^2$, let $F_{\beta_s}(\theta) = f_{\beta_s}(\theta, \mathcal{D})$, $r_s(\theta) = \frac{1}{2\lambda}\|\theta - \theta^s\|^2$. Due to the ρ-weak-convexity of $f(\theta)$, the $\frac{1}{\lambda}$-strong-convexity of $r_s(\theta)$ and the $\frac{\sqrt{d}L}{\beta_s}$-smoothness of $f_{\beta_s}(\theta)$, we have the following three inequality

$$\begin{aligned} F_{\beta_s}(\theta) &\geq F_{\beta_s}(\theta_t) + \langle \nabla F_{\beta_s}(\theta_t), \theta - \theta_t \rangle - \frac{\rho}{2}\|\theta_t - \theta\|^2 \\ r_s(\theta) &\geq r_s(\theta_{t+1}) + \langle \partial r_s(\theta_{t+1}), \theta - \theta_{t+1} \rangle + \frac{1}{2\lambda}\|\theta - \theta_{t+1}\|^2 \\ F_{\beta_s}(\theta_t) &\geq F_{\beta_s}(\theta_{t+1}) - \langle \nabla F_{\beta_s}(\theta_t), \theta_{t+1} - \theta_t \rangle - \frac{\sqrt{d}L}{2\beta_s}\|\theta_t - \theta_{t+1}\|^2 . \end{aligned} \tag{19}$$

Table 11: Experiments on RoBERTa-large(350M parameters).

Learning Rate η	$1e-5$	$2e-5$	$4e-5$	$6e-5$	$8e-5$	$1e-4$	$2e-4$	$4e-4$
$r = 0.5\%$	—	—	—	—	—	76.8	**78.6**	60.9
$r = 1\%$	—	—	—	76.5	78.8	**80.4**	69.3	—
$r = 2\%$	—	—	—	79.2	79.8	**80.4**	72.4	—
$r = 5\%$	—	—	75.0	**79.6**	78.6	73.2	52.1	—
$r = 10\%$	70.2	75.0	**79.5**	78.6	—	—	—	—
$r = 100\%$	74.8	**78.9**	78.2	—	—	—	—	—

combing them together, we have

$$\begin{aligned}
&F_{\beta_s}(\theta_{t+1}) + r_s(\theta_{t+1}) - (F_{\beta_s}(\theta) + r_s(\theta)) \\
\leq&\langle \nabla F_{\beta_s}(\theta_t) + \partial r_s(\theta_{t+1}), \theta_{t+1} - \theta \rangle \\
&+ \frac{\rho}{2}\|\theta_t - \theta\|^2 + \frac{\sqrt{d}L}{2\beta_s}\|\theta_t - \theta_{t+1}\|^2 - \frac{1}{2\lambda}\|\theta - \theta_{t+1}\|^2 .
\end{aligned} \tag{20}$$

If we set the gradient in θ_{t+1} to 0, there exists $\partial r_s(\theta_{t+1})$ such that

$$\partial r_s(\theta_{t+1}) = \frac{1}{\eta_s}(\theta_t - \theta_{t+1}) - \nabla \hat{f}_{\beta_s}(\theta_t, \xi_{t+1}). \tag{21}$$

where $\nabla \hat{f}_{\beta_s}(\theta_t, \xi_{t+1}) = \frac{1}{P}\sum_{p=1}^{P} \frac{1}{m}\sum_{i=1}^{m} \mathrm{CLIP}\left(\nabla f^p_{\beta_s}(\theta_t, x^i_{t+1})\right) + \mathbf{z}^p_t$ and $\mathbf{z}^p_t = z^p_t \cdot \mathbf{v}^p_t \left(z^p_t \sim N(0, \sigma^2 C^2)\right)$. Plugging the above equation into 20 and setting $\hat{\theta}_{t+1}$ be the updated parameter on dataset $\mathcal{D}$ without DP noise in iteration $t+1$. Since η is decreasing and β is increasing, $\eta_s \leq \frac{\beta_s}{\sqrt{d}L}$ holds true for all $s \in [0, S]$ when $\eta_1 \leq \frac{\beta_1}{\sqrt{d}L}$, by taking $\eta_s \leq \frac{\beta_s}{\sqrt{d}L}$, we have

$$\begin{aligned}
&F_{\beta_s}(\theta_{t+1}) + r_s(\theta_{t+1}) - (F_{\beta_s}(\theta) + r_s(\theta)) \\
\leq&\langle \nabla F_{\beta_s}(\theta_t) - \nabla \hat{f}_{\beta_s}(\theta_t, \xi_{t+1}), \theta_{t+1} - \theta \rangle + \langle \frac{1}{\eta_s}(\theta_t - \theta_{t+1}), \theta_{t+1} - \theta \rangle \\
&+ \frac{\rho}{2}\|\theta_t - \theta\|^2 + \frac{\sqrt{d}L}{2\beta_s}\|\theta_t - \theta_{t+1}\|^2 - \frac{1}{2\lambda}\|\theta - \theta_{t+1}\|^2 \\
=&\langle \nabla F_{\beta_s}(\theta_t) - \nabla \hat{f}_{\beta_s}(\theta_t, \xi_{t+1}), \theta_{t+1} - \hat{\theta}_{t+1} + \hat{\theta}_{t+1} - \theta \rangle \\
&+ \left(\frac{1}{2\eta_s} + \frac{\rho}{2}\right)\|\theta_t - \theta\|^2 - \left(\frac{1}{2\eta_s} + \frac{1}{2\lambda}\right)\|\theta - \theta_{t+1}\|^2 \\
=&\langle \nabla F_{\beta_s}(\theta_t) - \nabla \hat{f}_{\beta_s}(\theta_t, \xi_{t+1}), \hat{\theta}_{t+1} - \theta \rangle \\
&+ \eta_s \left\|\nabla F_{\beta_s}(\theta_t) - \nabla \hat{f}_{\beta_s}(\theta_t, \xi_{t+1})\right\|^2 + \left(\frac{1}{2\eta_s} + \frac{\rho}{2}\right)\|\theta_t - \theta\|^2 \\
&- \left(\frac{1}{2\eta_s} + \frac{1}{2\lambda}\right)\|\theta - \theta_{t+1}\|^2 .
\end{aligned} \tag{22}$$

Taking expectation on both sides, we have

$$\begin{aligned}
&\mathbb{E}\left[\phi_s(\theta_{t+1}) - \phi_s(\theta)\right] \leq \eta_s \mathbb{E}\left[\left\|\nabla F_{\beta_s}(\theta_t) - \nabla \hat{f}_{\beta_s}(\theta_t, \xi_{t+1})\right\|^2\right] \\
&+ \left(\frac{1}{2\eta_s} + \frac{\rho}{2}\right)\|\theta_t - \theta\|^2 - \left(\frac{1}{2\eta_s} + \frac{1}{2\lambda}\right)\|\theta - \theta_{t+1}\|^2 .
\end{aligned} \tag{23}$$

In the following proof, we will bound the term $\mathbb{E}\left[\left\|\nabla F_{\beta_s}(\theta_t) - \nabla \hat{f}_{\beta_s}(\theta_t, \xi_{t+1})\right\|^2\right]$ as follow:

$$\begin{aligned}
&\mathbb{E}\left[\left\|\nabla F_{\beta_s}(\theta_t) - \nabla \hat{f}_{\beta_s}(\theta_t, \xi_{t+1})\right\|^2\right] \\
=&\mathbb{E}\left[\left\|\nabla F_{\beta_s}(\theta_t) - \frac{1}{P}\sum_{p=1}^{P}\frac{1}{m}\sum_{i=1}^{m}\mathrm{CLIP}\left(\nabla f^p_{\beta_s}(\theta_t, x^i_{t+1})\right) + \mathbf{z}^p_t\right\|^2\right] \\
=&\frac{1}{P^2}\sum_{p=1}^{P}\mathbb{E}\left[\left\|\nabla F_{\beta_s}(\theta_t) - \frac{1}{m}\sum_{i=1}^{m}\mathrm{CLIP}\left(\nabla f^p_{\beta_s}(\theta_t, x^i_{t+1})\right)\right\|^2\right] \\
&+ \frac{dc_2^2 C^2 PT\log(P/\delta)}{\varepsilon^2 n^2} \\
=&\left\|\mathbb{E}_{\mathbf{v}}\left[\nabla F_{\beta_s}(\theta_t)\right] - \frac{1}{P}\sum_{p=1}^{P}\frac{1}{m}\sum_{i=1}^{m}\mathrm{CLIP}\left(\nabla f^p_{\beta_s}(\theta_t, x^i_{t+1})\right)\right\|^2 \\
&+ \frac{dc_2^2 C^2 PT\log(P/\delta)}{\varepsilon^2 n^2} .
\end{aligned} \tag{24}$$

Using the elementary inequality $(a-b)^2 \leq 2a^2 + 2b^2$, we have

$$\begin{aligned}
&\mathbb{E}\left[\left\|\frac{1}{2\beta}\left(f(\theta_{t-1} + \beta\mathbf{v}, x) - f(\theta_{t-1} - \beta\mathbf{v}, x)\right)\right\|^2\right] \\
=&\frac{1}{4\beta^2}\mathbb{E}[|f(\theta_{t-1} + \beta\mathbf{v}, x) - \mathbb{E}_{\mathbf{v}}[f(\theta_{t-1} + \beta\mathbf{v}, x)] \\
&+ \mathbb{E}_{\mathbf{v}}[f(\theta_{t-1} + \beta\mathbf{v}, x)] - f(\theta_{t-1} - \beta\mathbf{v}, x)|^2] \\
\leq&\frac{1}{2\beta^2}\mathbb{E}\left[\|f(\theta_{t-1} + \beta\mathbf{v}, x) - \mathbb{E}_{\mathbf{v}}[f(\theta_{t-1} + \beta\mathbf{v}, x)]\|^2\right] \\
&+ \frac{1}{2\beta^2}\mathbb{E}\left[\|\mathbb{E}_{\mathbf{v}}[f(\theta_{t-1} + \beta\mathbf{v}, x)] - f(\theta_{t-1} - \beta\mathbf{v}, x)\|^2\right] .
\end{aligned} \tag{25}$$

Since $\mathbf{v}$ has a symmetric distribution around the origin, we have

$$\begin{aligned}
&\mathbb{E}\left[\|f(\theta_{t-1} + \beta\mathbf{v}, x) - \mathbb{E}_{\mathbf{v}}[f(\theta_{t-1} + \beta\mathbf{v}, x)]\|^2\right] \\
=&\mathbb{E}\left[\|f(\theta_{t-1} - \beta\mathbf{v}, x) - \mathbb{E}_{\mathbf{v}}[f(\theta_{t-1} + \beta\mathbf{v}, x)]\|^2\right] .
\end{aligned} \tag{26}$$

Define $h(\theta) = f(\theta + \beta\mathbf{v})$. Since f is L-Lipschitz and $\mathbf{v} \in \mathcal{R}^d$ is sampled from standard Gaussian distribution. Then, by

[[41] Proposition 3.2], we have

$$\mathbb{P}\left(|h(\theta)-\mathbb{E}[h(\theta)]|\geq c\right)\leq 2e^{-\frac{c^2}{2\beta^2L^2}}. \tag{27}$$

Then, we have

$$\begin{aligned}&\mathbb{E}\left[\left(h(\theta)-\mathbb{E}[h(\theta)]\right)^2\right]=\int_0^{+\infty}\mathbb{P}\left(|h(\theta)-\mathbb{E}[h(\theta)]|^2\geq c\right)dc\\&=\int_0^{+\infty}\mathbb{P}\left(|h(\theta)-\mathbb{E}[h(\theta)]|\geq\sqrt{c}\right)dc\leq 2\int_0^{+\infty}e^{-\frac{c}{2\beta^2L^2}}dc\\&=4\beta^2L^2.\end{aligned} \tag{28}$$

By the definition of h, we have

$$\mathbb{E}\left[\left\|\frac{1}{2\beta}\left(f(\theta_{t-1}+\beta\mathbf{v},x)-f(\theta_{t-1}-\beta\mathbf{v},x)\right)\right\|^2\right]\leq 4L^4. \tag{29}$$

Furthermore, we can obtain that

$$\begin{aligned}&\mathbb{E}\left[\left\|\nabla f_\beta(\theta)-\mathbb{E}_{\mathbf{v}}\left[\nabla f_\beta(\theta)\right]\right\|^2\right]\\&\leq 2d\mathbb{E}\left[\left\|\frac{1}{2\beta}\left(f(\theta+\beta\mathbf{v})-\mathbb{E}_{\mathbf{v}}\left[f(\theta+\beta\mathbf{v})\right]\right)\right\|^2\right]\\&\quad+2d\mathbb{E}\left[\left\|\frac{1}{2\beta}\left(f(\theta-\beta\mathbf{v})-\mathbb{E}_{\mathbf{v}}\left[f(\theta-\beta\mathbf{v})\right]\right)\right\|^2\right]\leq 64d\beta^2L^4.\end{aligned} \tag{30}$$

Following assumption 1 that $\mathbb{E}\left[\left\|\nabla F_\beta(\theta)-\nabla f_\beta(\theta,x)\right]\right\|^2\leq\gamma^2$

$$\begin{aligned}&\left\|\mathbb{E}_{\mathbf{v}}\left[\nabla F_{\beta_s}(\theta_t)\right]-\frac{1}{P}\sum_{p=1}^{P}\frac{1}{m}\sum_{i=1}^{m}\mathrm{CLIP}\left(\nabla f^p_{\beta_s}(\theta_t,x^i_{t+1})\right)\right\|^2\\&\leq\frac{1}{P^2m^2}\sum_{p=1}^{P}\sum_{i=1}^{m}\left\|\nabla F^p_{\beta_s}(\theta_t)-\mathrm{CLIP}\left(\nabla f^p_{\beta_s}(\theta_t,x^i_{t+1})\right)\right\|^2\\&\quad+\frac{64d\beta_s^2L^4}{Pm}\\&\leq\frac{1}{P^2m^2}\sum_{p=1}^{P}\sum_{i=1}^{m}\left\|\nabla f^p_{\beta_s}(\theta_t,x^i_{t+1})-\mathrm{CLIP}\left(\nabla f^p_{\beta_s}(\theta_t,x^i_{t+1})\right)\right\|^2\\&\quad+\frac{64d\beta_s^2L^4}{Pm}+\frac{\gamma^2}{Pm}.\end{aligned} \tag{31}$$

Furthermore, we can bound the possibility of clipping happens

$$\mathbb{P}\left(\frac{1}{2\beta}\left|f(\theta_{t-1}+\beta\mathbf{v},x)-f(\theta_{t-1}-\beta\mathbf{v},x)\right|\geq C\right)\leq 2e^{-\frac{C^2}{2L^2}}. \tag{32}$$

We define Q^i_{t+1} to be the event that the clipping does not happen at iteration $t+1$ for sample x^i_{t+1}, and $\overline{Q^i_{t+1}}$ to be the event that the clipping does happen. When event Q^i_{t+1} happens, clipping does not happen at iteration $t+1$ for sample x^i_{t+1} such that $\left\|\nabla f_{\beta_s}(\theta_t,x^i_{t+1})-\mathrm{CLIP}\left(\nabla f_{\beta_s}(\theta_t,x^i_{t+1})\right)\right\|^2=0$, we can obtain that

$$\begin{aligned}&\mathbb{E}\left[\left\|\nabla f_{\beta_s}(\theta_t,x^i_{t+1})-\mathrm{CLIP}\left(\nabla f_{\beta_s}(\theta_t,x^i_{t+1})\right)\right\|^2\right]\\&=\left\|\nabla f_{\beta_s}(\theta_t,x^i_{t+1})-C\right\|^2\mathbb{P}(\overline{Q^i_{t+1}})\\&\leq\left(2dC^2+4dL^2\right)\cdot\exp\left(-\frac{C^2}{2L^2}\right).\end{aligned} \tag{33}$$

By setting $C^2=2L^2$, we have

$$\mathbb{E}\left[\left\|\nabla f_{\beta_s}(\theta_t,x^i_{t+1})-\mathrm{CLIP}\left(\nabla f_{\beta_s}(\theta_t,x^i_{t+1})\right)\right\|^2\right]\leq\frac{8dC^2}{e}. \tag{34}$$

Combining inequality 24, 31 and 34, we can bound the term $\mathbb{E}\left[\left\|\nabla F_{\beta_s}(\theta_t)-\nabla\hat{f}_{\beta_s}(\theta_t,\xi_{t+1})\right\|^2\right]$ by

$$\begin{aligned}&\mathbb{E}\left[\left\|\nabla F_{\beta_s}(\theta_t)-\nabla\hat{f}_{\beta_s}(\theta_t,\xi_{t+1})\right\|^2\right]\\&\leq+\frac{dc_2^2C^2PT\log(P/\delta)}{\varepsilon^2n^2}+\frac{64d\beta_s^2L^4}{Pm}+\frac{\gamma^2}{Pm}+\frac{8dC^2}{ePm}.\end{aligned} \tag{35}$$

Taking summation of the above inequality from $t=0$ to T_s-1 and by taking $\lambda<1/\rho$, we have

$$\begin{aligned}&\sum_{t=0}^{T_s-1}\phi_s(\theta_{t+1})-\phi_s(\theta)\leq\left(\frac{1}{2\eta_s}+\frac{\rho}{2}\right)\|\theta_0-\theta\|^2\\&+\eta_sT_s\left(\frac{dc_2^2C^2PT\log(P/\delta)}{\varepsilon^2n^2}+\frac{64d\beta_s^2L^4}{Pm}+\frac{\gamma^2}{Pm}+\frac{8dC^2}{ePm}\right)\\&-\left(\frac{1}{2\eta_s}+\frac{1}{2\lambda}\right)\|\theta-\theta_{T_s}\|^2.\end{aligned} \tag{36}$$

By employing Jensens' inequality, denoting the output of stage s by $\theta^s=\hat{\theta}_{T_s}=\frac{1}{T_s}\sum_{t=1}^{T_s}\theta_t$, since $\rho\leq 1/\lambda$ we can obtain that

$$\begin{aligned}&\phi_s(\hat{\theta}_{T_s})-\phi_s(\theta)\leq\left(\frac{1}{2\eta_sT_s}+\frac{1}{2T_s\lambda}\right)\|\theta_0-\theta\|^2\\&\qquad+\eta_s\left(\frac{dc_2^2C^2PT\log(P/\delta)}{\varepsilon^2n^2}+\frac{64d\beta_s^2L^4}{Pm}+\frac{\gamma^2}{Pm}+\frac{8dC^2}{ePm}\right).\end{aligned} \tag{37}$$

Then the proof for Lemma 5 is complete.

Theorem 5 *Suppose Assumption 1 holds and $f(\theta,x)$ is ρ-weakly-convex of θ. Then by setting $\lambda=\frac{3}{2\mu}$, $\eta_sT_s=\frac{3}{2\mu}$ and $\eta_s=\alpha_{s-1}\cdot\min\left\{\frac{Pm}{6\gamma^2},\frac{Pme}{48dC^2},\frac{\varepsilon^2n^2}{6dc_2^2C^2PT\log(P/\delta)},\frac{Pm}{384d\beta_s^2L^4}\right\}$, after $S=\lceil\log(\alpha_0/\alpha)\rceil$ stages, we have*

$$\phi_s(\theta^S)-\phi_s(\theta^*)\leq\alpha. \tag{38}$$

The total ZO oracle complexity is

$$\mathcal{O}\left(\left(\gamma^2+dC^2+d\beta_S^2L^4+\frac{dC^2P^2m\log(P/\delta)}{\varepsilon^2n^2}\right)\cdot\frac{1}{\mu\alpha}\right). \tag{39}$$

Proof for Theorem 2:

By taking $\theta=\theta^*$ in the above inequality and since , we have

$$\begin{aligned}\phi_s(\theta^s)-\phi_s(\theta^*)\le&\left(\frac{1}{4\eta_sT_s\mu}+\frac{1}{4T_s\lambda\mu}\right)\left(\phi_s(\hat{\theta}^{s-1})-\phi_s(\theta^*)\right)\\&+\eta_s\left(\frac{dc_2^2C^2PT\log(P/\delta)}{\varepsilon^2n^2}+\frac{64d\beta_s^2L^4}{Pm}+\frac{\gamma^2}{Pm}+\frac{8dC^2}{ePm}\right)\\\le&\left(\frac{1}{4\eta_sT_s\mu}+\frac{1}{4T_s\lambda\mu}\right)\alpha_{s-1}\\&+\eta_s\left(\frac{dc_2^2C^2PT\log(P/\delta)}{\varepsilon^2n^2}+\frac{64d\beta_s^2L^4}{Pm}+\frac{\gamma^2}{Pm}+\frac{8dC^2}{ePm}\right).\end{aligned} \tag{40}$$

By setting $\eta_s=\alpha_{s-1}\cdot\min\left\{\frac{Pm}{6\gamma^2},\frac{Pme}{48dC^2},\frac{\varepsilon^2n^2}{6dc_2^2C^2PT\log(P/\delta)},\frac{Pm}{384d\beta_s^2L^4}\right\}$ and $\lambda=\frac{3}{2\mu}$, $\eta_sT_s=\frac{3}{2\mu}$, we have

$$\phi_s(\theta^s)-\phi_s(\theta^*)\le\alpha_s. \tag{41}$$

By induction, after $S=\lceil\log(\alpha_0/\alpha)\rceil$ stages, we have

$$\phi_s(\theta^S)-\phi_s(\theta^*)\le\alpha. \tag{42}$$

The total ZO oracle complexity is

$$\mathcal{O}\left(\left(\gamma^2+dC^2+d\beta_S^2L^4+\frac{dC^2P^2m\log(P/\delta)}{\varepsilon^2n^2}\right)\cdot\frac{1}{\mu\alpha}\right). \tag{43}$$

Theorem 6 *Suppose Assumption 1 holds on $\theta'\in\mathcal{R}^{r\cdot d}$ and loss function $h(\theta',x)$ is ρ-weakly-convex of θ'. Then by setting $\lambda=\frac{3}{2\mu}$, $\eta_sT_s=\frac{3}{2\mu}$ and $\eta_s=\alpha_{s-1}\cdot\min\left\{\frac{Pm}{6\gamma^2},\frac{Pme}{48dC^2},\frac{\varepsilon^2n^2}{6rdc_2^2C^2PT\log(P/\delta)},\frac{Pm}{384rd\beta_s^2L^4}\right\}$, after $S=\lceil\log(\alpha_0/\alpha)\rceil$ stages, we have*

$$\psi_s(\theta'^S)-\psi_s(\theta^*)\le\alpha. \tag{44}$$

The total ZO oracle complexity is

$$\mathcal{O}\left(\left(\gamma^2+rdC^2+rd\beta_S^2L^4+\frac{rdC^2P^2m\log(P/\delta)}{\varepsilon^2n^2}\right)\cdot\frac{1}{\mu\alpha}\right). \tag{45}$$

Algorithm 5 greatly reduces the total complexity by a factor of $1/r$ compared with Algorithm 1.

Proof for Theorem 4:

The proof of Theorem 4 is similar to Theorem 2 expect the dimension to be fine-tuned is reduce from d to $r\cdot d$. With proof processes remaining the same. We can obtain similar results that the total ZO oracle complexity is

$$\mathcal{O}\left(\left(\gamma^2+rdC^2+rd\beta_S^2L^4+\frac{rdC^2P^2m\log(P/\delta)}{\varepsilon^2n^2}\right)\cdot\frac{1}{\mu\alpha}\right). \tag{46}$$

replacing d with $r\cdot d$ compared with Theorem 2.

Table 12: Experiments on RoBERTa-large (350M parameters) with privacy budget ($\varepsilon = 2$ and 4).

	$\varepsilon = 2$					$\varepsilon = 4$				
Task	SST-2	SST-5	SNLI	MNLI	TREC	SST-2	SST-5	SNLI	MNLI	TREC
$r = 0.5\%$	88.5	46.5	70.3	70.1	83.4	92.2	**50.5**	78.6	71.4	81.0
$r = 1\%$	91.7	**48.8**	78.4	68.1	88.6	92.2	47.1	**80.4**	64.7	**88.2**
$r = 2\%$	88.4	47.1	78.2	**71.4**	**89.6**	91.6	47.7	**80.4**	**71.5**	87.0
$r = 5\%$	92.7	46.1	77.0	69.7	80.8	**93.0**	45.4	79.6	69.4	80.6
$r = 10\%$	92.3	45.0	**78.9**	68.3	87.0	92.5	46.5	79.5	69.6	87.6
$r = 100\%$	**92.9**	45.8	76.8	66.3	83.4	92.6	42.0	78.9	67.3	83.8

Table 13: Experiments on RoBERTa-large (350M parameters) with privacy budget ($\varepsilon = 8$).

Task	SST-2	SST-5	SNLI	MNLI	TREC
$r = 0.5\%$	92.2	**49.8**	**81.4**	71.6	85.4
$r = 1\%$	92.2	47.4	79.2	**73.1**	**90.8**
$r = 2\%$	91.8	49.4	78.9	68.6	85.4
$r = 5\%$	**93.2**	46.2	77.7	66.8	79.4
$r = 10\%$	92.1	46.2	78.9	67.3	88.8
$r = 100\%$	**93.2**	41.2	76.5	69.2	84.2