GradOT: Training-free Gradient-preserving Offsite-tuning for Large Language Models

Kai Yao[1,2*], **Zhaorui Tan**[3*], **Penglei Gao**[4], **Lichun Li**[2], **Kaixin Wu**[2], **Yinggui Wang**[2], **Yuan Zhao**[2], **Yixin Ji**[5], **Wei Wang**[2†], **Jianke Zhu**[1†],
[1]Zhejiang University [2]Ant Group [3]University of Liverpool
[4]Cleveland Clinic Lerner Research Institution [5]Soochow University
jiumo.yk@antgroup.com, raytan@liverpool.ac.uk

Abstract

The rapid growth of large language models (LLMs) with traditional centralized fine-tuning emerges as a key technique for adapting these models to domain-specific challenges, yielding privacy risks for both model and data owners. One promising solution, called offsite-tuning (OT), is proposed to address these challenges, where a weaker emulator is compressed from the original model and further fine-tuned with adapter to enhance privacy. However, the existing OT-based methods require high computational costs and lack theoretical analysis. This paper introduces a novel OT approach based on gradient-preserving compression, named GradOT. By analyzing the OT problem through the lens of optimization, we propose a method that selectively applies compression techniques such as rank compression and channel pruning, preserving the gradients of fine-tuned adapters while ensuring privacy. Extensive experiments demonstrate that our approach surpasses existing OT methods, both in terms of privacy protection and model performance. Our method provides a theoretical foundation for OT and offers a practical, training-free solution for offsite-tuning of large-scale LLMs.

1 Introduction

Recent years have witnessed the rapid growth of large language models (LLMs)(Devlin et al., 2019; Radford et al., 2019; Touvron et al., 2023; Du et al., 2022; Dong et al., 2024; Wies et al., 2024), whose centralized fine-tuning (Wortsman et al., 2022; Zhou et al., 2022; Wei et al., 2022; Ouyang et al., 2022; Tan et al., 2024; Liu et al., 2024a; Yao et al., 2024) is a common approach for adapting them to more complex, domain-specific tasks. However, the required co-location of the model and data raises risks of jeopardizing the privacy of both their owners (Chua et al., 2023; Xiao et al., 2023; Li et al., 2020; Zou et al., 2023; Ye et al., 2024; Liu et al., 2024b), posing significant barriers to LLMs' applications on sensitive downstream fine-tuning.

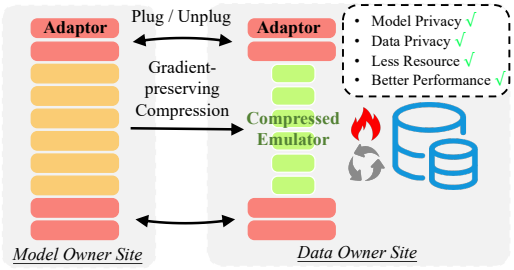


Figure 1: Illustration of our proposed Gradient-preserving Offsite-tuning (GradOT).

Offsite-tuning (OT) (Xiao et al., 2023) has emerged as a promising solution to safeguard the privacy of both data and model owners. Fig. 1 illustrates the process of our proposed method, which follows the general process of OT methods. As seen from this process diagram, OT-based methods usually involve lossy compression of the large language model into a smaller, weaker version, defined as an emulator on the model owner's side. The adapter is further fine-tuned on the compressed emulator with the data sent by data owners and will be returned to the model owner after fine-tuning, replacing the original adapters to form the fully adapted model.

There are two crucial objectives in OT-based fine-tuning: (1) *Privacy protecting* aims to intentionally increase the performance gap between the finetuned emulator and the fully finetuned model. (2) *Performance preserving* aims to, on the contrary, improve the performance of the final plug-in model. The development of a feasible lossy compression method that optimizes these two objectives is the key challenge in this field. OT (Xiao et al., 2023) attempted to achieve offsite-tuning by leveraging LayerDrop (Sajjad et al., 2023) and knowledge distillation (Sanh et al., 2019; Hinton

[*]These authors contributed equally to this work.
[†]Corresponding authors.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 5115–5130
July 27 - August 1, 2025

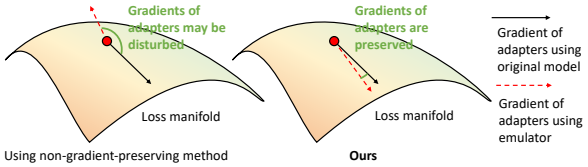


Figure 2: Diagram of gradient preserving of adapters through emulators.

et al., 2015), which requires high computational cost and hinders its application in practices on large-scale LLMs. Following this, CRaSh (Zhang et al., 2023a) used repetitive shared layers to replace dropped layers, proposing a novel training-free OT method. In addition, another noteworthy method termed ScaleOT (Yao et al., 2025) proposed to estimate the layer-wise importance with reinforcement learning and replace the less important layers with lightweight modules, significantly improving the privacy protection and final fine-tuned model performance. These OT-based approaches not only ensure privacy for both the data and the model but also encourage the use of the fully adapted model, fostering a mutually beneficial relationship between the data and model owners. However, they heavily rely on empirical validation and lack theoretical analysis, which limits the development of OT-based methods and their practical applications.

In this paper, we take the first step in analyzing the OT problem from an optimization perspective. We formally define the OT objective, showing that the problem centers on maximizing privacy while minimizing the gradient discrepancies between the adapters and the compressed emulator, which has not yet been explored in this field. By approximating this formal objective, we introduce a gradient-preserving compression score, which is used to construct the emulator. As shown in Fig. 2, our gradient-preserving compression score (GSC)-based framework coordinates adapter optimization through joint training with the original model and compressed emulators, achieving dual objectives of maintaining task performance in the final plug-in model and enforcing privacy preservation via gradient consistency mechanisms. As shown in Fig. 2, our gradient-preserving compression score (GSC)-based method maximizes the adapters that are trained with the original model and emulators, ensuring the performance of the final plug-in model. Then, we propose a novel training-free approach called gradient-preserving offsite tuning (GradOT) as shown in Fig. 1, which focuses on preserving the gradients of adapters to enhance privacy in the context of OT. Inspired by Yao et al. (2025), our method applies different compression techniques to different blocks within the LLM. Specifically, we use Dynamic Rank Decomposition (DRD) for the MHA and Selective Channel Pruning (SCP) for the MLP. Importantly, our compression is performed based on the scores, ensuring gradient preservation while allowing for increased loss, which enhances both performance and privacy.

Extensive experiments demonstrate that our method outperforms previous approaches. Our contributions are as follows:

- We formally define the OT problem and present its tractable objective. Based on the objectives, we derive the gradient-preserving compression score for OT tasks.
- Integrating the gradient-preserving compression score, we propose a novel training-free OT method named GradOT, which consists of Dynamic Rank Decomposition (DRD) and Selective Channel Pruning (SCP).
- Extensive experiments on various LLMs and datasets showcase the effectiveness of our GradOT, validating the efficacy of our gradient-preserving compression score.

2 Related Work

Large Language Models. As models grow in size and complexity, pre-trained Large Language Models (LLMs) have demonstrated impressive performance across various natural language processing (NLP) tasks (Touvron et al., 2023; Brown et al., 2020). These models possess extensive general knowledge, enabling them to effectively address unknown problems through zero-shot learning or in-context learning (Li et al., 2020; Zou et al., 2023). Nonetheless, when dealing with complex problems, transfer learning with a small dataset remains the preferred choice, as it avoids costly retraining and leverages pretrained knowledge for better performance. Offsite tuning, a subset of transfer learning, aims to utilize the knowledge of LLMs for downstream tasks while ensuring bidirectional privacy between the model and the data sites.

Offsite-tuning. In contrast to traditional privacy approaches that emphasize data, such as federated learning (Nguyen et al., 2021) and differential privacy (Singh et al., 2024), offsite tuning methods (Hong et al., 2024; Xiao et al., 2023) prioritize model privacy. Previous research has pri-

marily relied on empirical experiments. For example, OT (Xiao et al., 2023) determined through experiments that LayerDrop meets offsite tuning objectives, whereas pruning and quantization are challenging to implement. Building on this, CRaSh (Zhang et al., 2023a) enhanced performance by sharing remaining layers after layer dropping. ScaleOT (Yao et al., 2025) introduced layerwise importance to preserve critical layers and replace less significant ones with lightweight modules. Despite their efficacy, the absence of theoretical analysis has constrained further development. Furthermore, analyzing layer-based methods is challenging due to significant structural differences among various models. This paper offers a theoretical analysis from the perspective of weights and provides a universally effective indicator tailored for offsite tuning.

3 Method

3.1 Theoretical Motivation

We address privacy concerns that hinder the sharing and co-location of data and LLMs between their respective owners, aiming to tune the model using the data owner's data while avoiding access to the model owner's weights. The goal of offsite-tuning is to address the privacy concerns of both data and model owners by indirectly sharing original data and models. Typically, the model owner offers the adapters with an emulator that is compressed from the original model middleweights for data owners to fine-tune on their side. Those tuned adapters would be returned and combined with the original model's middleweights as the final tuned model.

Following previous works (Yao et al., 2025), the goal of offsite tuning is to find a lossy compression function $\mathcal{F}$ that compresses a full model $\mathcal{M}$ into an alternative, smaller, and weaker model $\widehat{\mathcal{M}}$. Similar to previous works (Xiao et al., 2023; Zhang et al., 2023a) and for simplification, we denote $\mathcal{M} = \mathcal{A}_2 \circ \mathcal{E} \circ \mathcal{A}_1(\cdot)$, where $\mathcal{A} = [\mathcal{A}_1, \mathcal{A}_2]$ denotes the trainable top and bottom layers of full model, and $\mathcal{E}$ represents the frozen middle layers. The compressed version $\widehat{\mathcal{M}} = \widehat{\mathcal{A}}_2 \circ \widehat{\mathcal{E}} \circ \widehat{\mathcal{A}}_1(\cdot)$ where $\widehat{\mathcal{E}} = \mathcal{F}(\mathcal{E}), \widehat{\mathcal{A}}_2 = \mathcal{A}_2, \widehat{\mathcal{A}}_1 = \mathcal{A}_1$ is used as the initialization for the fine-tuning at the data owner side. Then, the data owner fine-tunes the adapters with $\widehat{\mathcal{E}}$ resulting in the *Emulator Fine-tuned* model denoted as $\widehat{\mathcal{M}}' = \widehat{\mathcal{A}}'_2 \circ \widehat{\mathcal{E}} \circ \widehat{\mathcal{A}}'_1(\cdot)$. The trained adapters $\widehat{\mathcal{A}}'_1, \widehat{\mathcal{A}}'_2$ are then returned to the model owner to merge, combining as the final *Plug-in* model $\mathcal{M}^* = \widehat{\mathcal{A}}'_2 \circ \mathcal{E} \circ \widehat{\mathcal{A}}'_1(\cdot)$. Similarly, we also denote the directly fine-tuned $\mathcal{M}$ as $\mathcal{M}' = \mathcal{A}'_2 \circ \mathcal{E} \circ \mathcal{A}'_1(\cdot)$, where $\mathcal{A}'_1, \mathcal{A}'_2$ are the fine-tuned adapters with original $\mathcal{E}$.

Following the method of Lagrange multiplier, given a downstream dataset $\mathcal{D}$ that consists of training D_{train} and testing D_{test} sets, the objective of offsite tuning is to find $\mathcal{F}$ to minimize the test loss gap between $\mathcal{M}^*$ and $\mathcal{M}'$ while maximize the loss discrepancy between $\mathcal{M}^*$ and $\widehat{\mathcal{M}}'$:

$$\min_{\mathcal{F}} \underbrace{\mathcal{L}(\mathcal{M}^*, D_{test}) - \mathcal{L}(\mathcal{M}', D_{test})}_{\text{Term (1): Plug-in performance preserving}} \underbrace{-\lambda \Big(\mathcal{L}(\mathcal{M}^*, D_{test}) - \mathcal{L}(\widehat{\mathcal{M}}', D_{test})\Big)}_{\text{Term (2): Privacy protecting}} \quad (1)$$
$$\text{s.t.} \underbrace{\min_{\widehat{\mathcal{A}}'_1, \widehat{\mathcal{A}}'_2} \mathcal{L}(\widehat{\mathcal{M}}', D_{train}), \min_{\mathcal{A}'_1, \mathcal{A}'_2} \mathcal{L}(\mathcal{M}', D_{train})}_{\text{Conditions}}$$

where $\mathcal{L}(\cdot, \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{(x_i, y_i) \in \mathcal{D}} \ell(\cdot, x_i, y_i)$ is the average task loss of any given model on the dataset, λ controls the privacy and utility trade-off of the offsite-tuning method, ℓ denotes the task loss function.

In this article, we explore the theoretical aspects of compressing models to achieve the goals of offsite tuning. In the following sections, we will first present our theoretical foundation and then propose methods to align with this framework.

3.2 Forming Objectives of OT through the Perspective of Gradient

This paper seeks an approach that meets Eq. (1) of offsite tuning from the perspective of model weights and optimization. Specifically, we can regard the compression of weights as adding the noise δ to the weights, i.e., $\widehat{w} = w + \delta$. Given model with n layers, $\mathcal{M} = \{m_1, m_2, \ldots, m_n\}$, the loss of this model is:

$$\ell(w, x_j, y_j) = L(m_n(m_{n-1}(\cdots m_1(x_j, w_1) \cdots, w_{n-1}), w_n), y_j), \quad (2)$$

where $W = \{w_1, w_2, \cdots, w_n\}$ denote the weight of each layer of M in order. Considering the compressed weight $\widehat{W} = \{\hat{w}_1, \hat{w}_2, \cdots, \hat{w}_n\}$, we only compress the weights of middle layers to generate the emulator, and the loss of emulator $\widehat{\mathcal{M}}$ is:

$$\ell(\hat{w}, x_j, y_j) = L(m_n(m_{n-1}(\cdots m_1(x_j, \hat{w}_1) \cdots, \hat{w}_{n-1}), \hat{w}_n), y_j),$$
$$\hat{w}_i = \begin{cases} w_i + \delta_i & \text{if } n_1 \le i \le n_2 \\ w_i & \text{otherwise} \end{cases}, \quad (3)$$

where n_1 and n_2 are the start and end layer index of the emulator.

Considering Term (1), the main problem is figuring out how much the gradients of the i^{th} layer's weight w_i would change while adding δ to the original model weights, which can be formed as:

$$\min \nabla_{w_i} \ell(\hat{w}) - \nabla_{w_i} \ell(w). \tag{4}$$

In practice, it is almost impossible to estimate the gradient variations of the adapters through all layers of the emulator simultaneously due to memory limitations and computational cost. Based on the Chain Rule, we minimize the gradient variations of each layer, which ultimately presses the gradients of the adapters. Therefore, focusing on the gradient changes for a given weight w_i that denoted as $\nabla_{w_i} \ell(w)$, while applying δ_i, we have the following according to Taylor expansion:

$$\begin{aligned} &\nabla_{w_i} \ell(\hat{w}) = \nabla_{w_i} \ell(w + \delta) \\ &\approx \nabla_{w_i} \ell(w) + \frac{\partial^2 \ell}{\partial w_i^2} \delta_i + \text{higher-order terms}, \end{aligned} \tag{5}$$

where those higher-order terms can be omitted, and $\frac{\partial^2 \ell}{\partial w_i^2}$ is the partial hessian matrix of full model.

We then maximize the loss gap. To seek a tractable approximation of it, we need to expand $\ell(w)$ and $\ell(\hat{w})$ in the form of the *total differential*. We remind readers of the total differential of the model.

Theorem 3.1 *The sum of the products of each partial derivative and the corresponding small change in the weight variable can estimate the increment in the loss function at a given point.*

Theorem 3.1 is the expression of the concept of total differential in weight factorization in neural networks. Through the definition of total differential and Theorem 3.1, we can build the connections between $\ell(w)$ and $\ell(\hat{w})$. For the differentiable function ℓ with the small variable δ, the following equation is established:

$$\ell(\hat{w}) = \ell(w) + \sum_{i=n_1}^{n_2} \frac{\partial \ell}{\partial w_i} \odot \delta_i, \tag{6}$$

where $\odot$ is the inner product. Eq. (6) is obvious since $\hat{w}_i - w_i = \delta_i$ (see more details in Appendix A.1).

To achieve the goal of offsite tuning, a feasible solution is to ensure the gradient of the emulator is as close to raw as possible, but the divergence between the values of losses should be large:

$$\begin{aligned} &\min_{\delta \in \Delta} |\nabla_{w_i} \ell(\hat{w}) - \nabla_{w_i} \ell(w)| \\ &\qquad - \lambda(\ell(\hat{w}) - \ell(w)), \\ &\approx \underbrace{\sum^{n} || \frac{\partial^2 \ell}{\partial w_i^2} \delta_i ||_1}_{\text{For Eq. (1) Term (1)}} \underbrace{-\lambda(\sum^{n} \frac{\partial \ell}{\partial w_i} \odot \delta_i)}_{\text{For Eq. (1) Term (2)}}. \end{aligned} \tag{7}$$

3.3 Deriving OT method through Training-free Compression

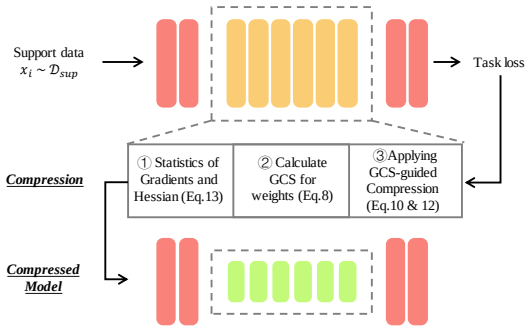
This section demonstrates how to achieve Eq. (7) empirically. We concentrate on the design of offsite-tuning for the Transformer architecture (Vaswani et al., 2017), which is extensively used in LLMs (Radford et al., 2019; Brown et al., 2020; Touvron et al., 2023). A typical Transformer (Vaswani et al., 2017) layer is composed of two blocks: the Multi-Head Attention (MHA) and the Multilayer Perceptron (MLP). The MHAs facilitate interactions among tokens, while the MLPs further process information transformation within tokens. As the overall method shown in Fig. 3 (a), we specifically employ Dynamic Rank Decomposition (DRD) on MHA and Selective Channel Pruning (SCP) on MLP as illustrated in Fig. 3 (b). Both compression methods rely on Eq. (7) to achieve optimal offsite-tuning. For each compressed weight $\widehat{w}_i$, we compute the Gradient-preserving Compression Score (GCS) for weight noise δ_i as follows:

$$\text{GCS}(\delta_i) = \underbrace{|| \frac{\partial^2 \ell}{\partial w_i^2} \delta_i ||_1}_{\text{Score Term (1)}} \underbrace{-\lambda \frac{\partial \ell}{\partial w_i} \odot \delta_i}_{\text{Score Term (2)}}. \tag{8}$$

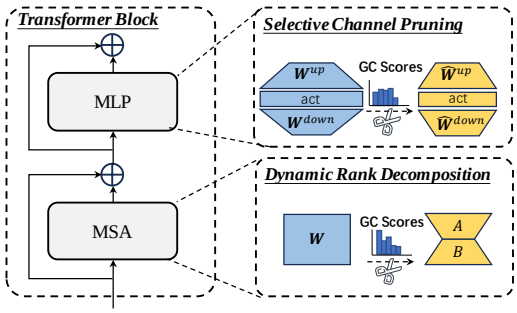
where the Score Term (1) and (2) correspond to the Eq. (1) Term (1) and (2), respectively. Our goal is to determine δ_i by minimizing the scores while adhering to the given compression ratio. We set different trade-off factors λ_{mha} and λ_{mlp} for MHA and MLP, respectively.

Dynamic Rank Decomposition. Rank compression is an effective matrix compression strategy, which reduces the size of a weight matrix by decomposing it into the product of two low-rank matrices. Considering the weight of a linear layer, denoted as $\boldsymbol{W} \in \mathbb{R}^{d_o \times d_i}$, we utilize truncated Singular Value Decomposition (SVD) (Eckart and Young, 1936) as:

$$\boldsymbol{W} = \boldsymbol{U} \boldsymbol{\Sigma} \boldsymbol{V}^T. \tag{9}$$



(a) Overview of offsite-tuning oriented compression



(b) Details of GCS-guided compression

Figure 3: Overview of our Gradient-preserving Compression Score (GCS) guided compression strategy.

For each weight in the MHA, given the compression ratio $\boldsymbol{r}_{mha}$, we generate the compressed weight as follow,

$$\widehat{\boldsymbol{W}} = \boldsymbol{B}\boldsymbol{A}, \boldsymbol{A} = \boldsymbol{V}^T_{s,:}, \boldsymbol{B} = \boldsymbol{U}_{:,s}\boldsymbol{\Sigma}_{s,s},$$
$$\text{s.t.} \min_s \text{GCS}(\widehat{\boldsymbol{W}} - \boldsymbol{W}), |s| = \frac{\boldsymbol{r}_{mha} d_i d_o}{d_i + d_o}, \quad (10)$$

where $\boldsymbol{A} \in \mathbb{R}^{|s| \times d_i}$, $\boldsymbol{B} \in \mathbb{R}^{d_o \times |s|}$, s is the set of the rank index containing $|s|$ elements, $\boldsymbol{\Sigma}_{s,s}$ contains the selected singular values, and $\boldsymbol{U}_{:,s}$ and $\boldsymbol{V}_{s,:}$ are the corresponding singular vectors. To ensure accurate scoring, we consistently select the top 5% of ranks due to the constraint that δ_i should not be excessively large.

Selective Channel Pruning. For expressive capability, the MLP's intermediate dimension is empirically set to be very high, significantly exceeding its input and output dimensions. Formally, the formulation of MLP blocks is as follows:

$$f_{\text{MLP}}(\boldsymbol{X}) = \sigma(\boldsymbol{X}\boldsymbol{W}^{up})\boldsymbol{W}^{down}, \quad (11)$$

where $\boldsymbol{W}^{up} \in \mathbb{R}^{d_h \times d_{int}}$, $\boldsymbol{W}^{down} \in \mathbb{R}^{d_{int} \times d_h}$, σ is the activation function, d_h and d_{int} denote hidden dimension and intermediate dimension of LLM respectively, $d_h \ll d_{int}$. Similar to Eq. 10, given compression ratio $\boldsymbol{r}_{mlp}$, we search for the optimal subset s within the intermediate dimension to reduce parameters:

$$\widehat{f_{\text{MLP}}}(\boldsymbol{X}) = \sigma(\boldsymbol{X}\boldsymbol{W}^{up}_{:,s})\boldsymbol{W}^{down}_{s,:},$$
$$\text{s.t.} \min_s [\text{GCS}(\boldsymbol{W}^{up}_{:,s} - \boldsymbol{W}^{up}) + \quad (12)$$
$$\text{GCS}(\boldsymbol{W}^{down}_{s,:} - \boldsymbol{W}^{down})], |s| = \boldsymbol{r}_{mlp} d_{int}.$$

Approximation of Hessian Matrix. In our score, we have to calculate the partial Hessian of each linear in the emulator. Directly computing a linear's partial Hessian matrix demands substantial computational resources, particularly for LLMs. (Radhakrishnan et al., 2024) also highlights that the features extracted by a given neural network layer are proportional to the average gradient outer product with respect to the input of that layer. Consequently, one feasible solution (Kunstner et al., 2019) is to estimate the Hessian using Fisher Information. Nonetheless, estimating the entire Hessian matrix remains exceptionally challenging because the full Hessian or Fisher matrix is a $P \times P$ matrix, where P represents the number of parameters.

Algorithm 1 Gradient-preserving Compression

Requires: The LLM $\mathcal{M}$ with n layers, the support dataset $\mathcal{D}_{sup}$, compression ratios $\boldsymbol{r}_{mha}$ and $\boldsymbol{r}_{mlp}$, trade-off factors λ_{mha} and λ_{mlp}.
for all $i \in \{1, \ldots, |\mathcal{D}_{\text{sup}}|\}$ **do**
  Compute the loss $\ell(w, x_i, y_i)$
  Perform backpropagation
  Accumulate gradients $\sum \frac{\partial \ell}{\partial w_i}$
  Accumulate approximate second-order gradients $\sum \frac{\partial^2 \ell}{\partial w_i^2}$ using Eq. (13)
Initialize Emulator $\widehat{\mathcal{E}}$ and Adapter $\widehat{\mathcal{A}}$
for all $w_i \in \widehat{\mathcal{E}}$ **do**
  if w_i belongs to MHA **then**
    $\widehat{w_i} = \text{DRD}(w_i, \boldsymbol{r}_{mha}, \lambda_{mha})$ based on Eq. (10)
  else if w_i belongs to MLP **then**
    $\widehat{w_i} = \text{SCP}(w_i, \boldsymbol{r}_{mlp}, \lambda_{mlp})$ based on Eq. (12)
return $\widehat{\mathcal{E}}, \widehat{\mathcal{A}}$

To mitigate this challenge, we utilize a Kronecker-factored approximation (KFAC) of individual weight matrices (Daxberger et al., 2021; Ritter et al., 2018; Yang et al., 2024), approximating the Fisher matrix through block structures within each linear layer. For the i_{th} linear layer, we compute the block by denoting the input as $\boldsymbol{a}_{i-1}$ and output as $\boldsymbol{b}_i$. Then, the partial Hessian can be esti-

Method	*Setting*	OBQA	PIQA	ARC-E	ARC-C	Hella.	SciQ	WebQs	RACE	Avg.	Δ ↑
Full Large Language Model											
Zero-shot (ZS)		23.4	71.6	56.9	23.5	41.5	84.4	4.6	34.2	42.5	-
Fine-tuning (FT)		31.4	75.2	61.3	27.7	42.7	92.5	31.2	37.0	49.9	
Post-Training Compression											
OT	Emu. ZS ↓	19.4	68.7	53.9	21.5	35.1	80.9	1.3	33.0	39.2	
	Emu. FT ↓	24.8	71.6	58.1	26.1	37.0	92.2	24.3	38.6	46.6	2.4
	Plug-in ↑	29.0	74.5	59.4	27.8	43.3	92.9	26.2	38.9	49.0	
ScaleOT	Emu. ZS ↓	17.2	63.1	41.8	19.5	32.1	59.9	0.1	27.2	**32.6**	
	Emu. FT ↓	27.2	70.9	52.5	26.5	37.8	90.0	25.3	39.1	**46.2**	**3.7**
	Plug-in ↑	28.2	75.2	61.9	28.3	42.9	94.0	28.2	40.8	**49.9**	
Training-free Compression											
OT†	Emu. ZS ↓	13.8	58.4	34.9	19.0	27.0	49.8	0.0	22.7	28.2	
	Emu. FT ↓	24.6	69.3	50.4	21.2	32.7	89.4	21.8	36.5	**43.2**	3.3
	Plug-in ↑	26.4	72.7	58.3	23.0	41.2	90.8	21.4	37.9	46.5	
CRaSh	Emu. ZS ↓	14.0	57.0	35.9	18.5	25.9	84.3	4.7	34.2	34.3	
	Emu. FT ↓	25.0	68.9	50.0	21.5	33.6	88.9	21.8	38.9	43.6	**4.8**
	Plug-in ↑	30.2	73.2	60.0	24.8	41.9	93.1	23.7	39.9	48.4	
GradOT	Emu. ZS ↓	12.2	55.8	33.8	18.0	26.9	49.4	0.0	22.3	**27.3**	
	Emu. FT ↓	27.0	70.2	51.1	23.8	33.6	88.2	28.4	38.7	45.0	**4.8**
	Plug-in ↑	30.8	73.6	61.3	27.6	41.7	93.9	29.2	40.6	**49.8**	

Table 1: Comparative results of offsite-tuning with OPT-1.3B on eight datasets. Δ denotes the performance difference between Emulator (Emu.) fine-tuning and Plug-in. † denotes OT without knowledge distillation. Best in **bold** and second best in underline.

mated by the following,

$$\frac{\partial^2 \ell}{\partial w_i^2} \approx \frac{1}{|\mathcal{D}|} \sum_{(x_i, y_i) \in \mathcal{D}} [(\boldsymbol{a}_{i-1} \boldsymbol{a}_{i-1}^T) \otimes (\boldsymbol{g}_i \boldsymbol{g}_i^T)], \quad (13)$$

where $\boldsymbol{g}_i = \frac{\partial \ell}{\partial \boldsymbol{b}_i}$ is the gradient of $\boldsymbol{b}_i$, $\otimes$ denotes Kronecker-factored multiplication.

Overall. By leveraging the proposed scoring for the offsite-tuning objective, we can effectively assess the impact of weight noise on both loss and gradient metrics by utilizing the gradients and second-order derivatives of the original network. This method demonstrates greater efficiency compared to traditional approaches. Consequently, we introduce a support dataset $\mathcal{D}_{sup}$ to compute relevant values necessary for the score computation. Notably, for an identical network, we only need to perform the statistical analysis once, thereby substantially reducing computational demands and time, as shown in Appendix B.1. Additionally, during the compression phase, we sequentially compress each weight in the emulator and determine the optimal results based on the calculated scores, significantly enhancing both efficiency and performance, as detailed in Appendix A.2. Our method is detailed in Algorithm 1 for clarity.

4 Experiments

4.1 Experimental Setup

Datasets. We validate our method across eight commensense datasets following previous works (Yao et al., 2025), including Multi-Choice QA: OpenBookQA (Mihaylov et al., 2018), PIQA (Bisk et al., 2020), SciQ (Welbl et al., 2017), RACE (Lai et al., 2017); Closed-Book QA: ARC-Easy/Challenge (Clark et al., 2018); and Sentence Completion: HellaSwag (Zellers et al., 2019). We use `lm-eval-harness` [*] to evaluate LLMs for a fair comparison. In order to preserve the privacy of the target datasets, we adhere Zhang et al. (2023a) to generate the supporting dataset from the same tasks, resulting in a total of 1,500 samples across BoolQ (Clark et al., 2019), TriviaQA (Joshi et al., 2017), and CoPA (Wang et al., 2019).

Models. We evaluate our method on large language models, including OPT-1.3B (Zhang et al., 2023b), OPT-6.7B (Zhang et al., 2023b), LLaMA-7B, and LLaMA-13B (Touvron et al., 2023). In line with previous studies (Xiao et al., 2023), we consistently select the first and last two layers as adapters, meaning that only about 10% of the parameters are fine-tuned, as opposed to full fine-tuning. The remain-

[*] https://github.com/EleutherAI/lm-evaluation-harness

Setting	OBQA	PIQA	ARC-E	ARC-C	HellaSwag	SciQ	WebQs	RACE	Avg.
OPT-6.7B									
Full Model ZS	27.6	76.2	65.5	30.6	50.5	90.1	8.8	38.2	48.4
Emulator ZS	19.6	68.8	53.6	24.7	38.1	83.0	2.4	31.6	40.2
Emulator FT	28.4	73.6	64.5	30.9	46.0	93.1	23.4	41.5	50.2
Post-training Compression									
ScaleOT Plug-in	33.2	78.1	70.1	35.3	52.2	95.7	33.9	45.3	**55.5**
Training-free Compression									
OT† Plug-in	33.8	77.7	66.8	33.9	52.1	91.9	23.9	44.1	53.0
CRaSh Plug-in	38.8	78.0	70.7	36.3	53.4	95.3	26.1	45.2	55.3
GradOT Plug-in	34.8	78.0	70.0	34.8	52.4	95.3	32.7	45.1	**55.4**
LLaMA-7B									
Full Model ZS	28.2	78.3	67.3	38.2	56.4	89.7	0.0	40.0	49.8
Emulator ZS	23.0	72.1	52.8	30.0	45.2	72.6	0.0	32.3	41.0
Emulator FT	30.6	76.3	63.3	32.6	49.4	91.8	29.6	43.2	52.1
Post-training Compression									
ScaleOT Plug-in	37.4	79.7	73.2	42.3	58.1	95.7	33.7	45.6	**58.2**
Training-free Compression									
OT† Plug-in	33.0	78.8	69.6	39.0	57.4	83.5	27.3	44.0	54.1
CRaSh Plug-in	34.6	80.0	71.3	41.8	58.4	95.1	29.8	45.6	57.1
GradOT Plug-in	33.8	80.5	72.8	41.7	57.9	95.4	38.3	44.8	**58.2**

Table 2: Comparative results of offsite-tuning with large-size language models on 8 question answering benchmarks.

Setting	OBQA	ARC-E	ARC-C	WebQs
Full Model ZS	30.6	74.5	43.9	0.0
Emulator ZS	24.0	51.8	30.2	0.0
Emulator FT	32.6	66.7	39.2	37.9
OT† Plug-in	34.4	76.5	43.8	35.4
GradOT Plug-in	**36.2**	**77.8**	**50.1**	**48.1**

Table 3: Comparative results of offsite-tuning on LLaMA-13B.

ing layers are then lossy compressed to generate the emulator. For compression, we set $[\boldsymbol{r}_{mha}, \boldsymbol{r}_{mlp}]$ to $[0.4, 0.7]$ for 1.5B model and $[0.5, 0.8]$ for 7B models, resulting in emulators with 60% and 70% parameters respectively. We present implementation details about experiments in Appendix A.3.

4.2 Main Results

We conduct a comprehensive comparison of our method's offsite tuning performance against existing state-of-the-art (SoTA) methods. For training-based methods, our comparison includes OT (Xiao et al., 2023) and ScaleOT (Yao et al., 2025), while for training-free methods, we compare with Uniform LayerDrop (achieved by OT without knowledge distillation) and CRaSh (Zhang et al., 2023a). Meanwhile, we present the results of full model zero-shot (ZS) and fine-tuning (FT), which act as the baseline for this study.

Table 1 presents the comparative results of offsite tuning on medium-sized models. All methods meet the offsite tuning conditions, meaning the plug-in's performance surpasses both the full model ZS and emulator FT. However, there remains a significant gap between the plug-in and full model FT performances, particularly for CRaSh and OT†. Nonetheless, our method achieves promising plug-in results, with performance only slightly lower than that of full model FT. On the other hand, when compared with the SoTA lossless offsite-tuning method, ScaleOT, which utilizes dense post-training compression, our method offers superior privacy protection. Notably, our approach achieves the lowest emulator ZS performance, highlighting the efficacy of the proposed scoring mechanism that amplifies task loss. Overall, our method delivers the best privacy-utility trade-off, making it a promising alternative to existing training-based methods.

We subsequently validated the effectiveness of our method when scaling up to larger models, including OPT-6.7B, LLaMA-7B and LLaMA-13B, as shown in Table 2 and Table 3. The results demonstrate that our proposed method exceeds previous training-free OT methods and achieves competitive plug-in performance compared to the existing post-training method, ScaleOT. Notably, we significantly outperform the training-free methods, OT and CRaSh, by an average of 4.1% and 1.1% on the LLaMA-7B model. Additionally, we consis-

DRD	SCP	r	OBQA		ARC-C		SciQ	
			P.in	Δ	P.in	Δ	P.in	Δ
-	-	1.0	31.4	0.0	27.7	0.0	92.5	0.0
-	✓	0.8	**34.0**	2.0	**29.5**	1.7	93.3	1.3
✓	-	0.8	31.0	3.4	28.3	3.4	93.1	2.6
✓	✓	0.6	30.8	**3.8**	27.6	3.8	**93.9**	**5.7**
w/o S	✓	0.6	30.0	-0.2	28.9	4.0	93.8	4.2
✓	w/o S	0.6	29.8	1.0	25.8	**4.4**	93.0	3.5

Table 4: Ablation study. r denotes the overall parameter ratio of the emulator $\widehat{\mathcal{E}}$ maintained from the original $\mathcal{E}$, while w/o S represents the compression without our proposed score.

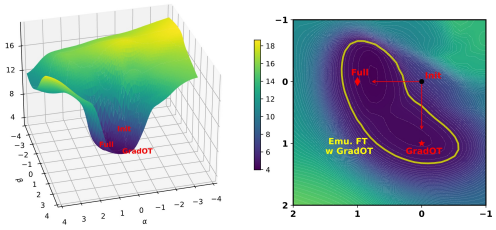


Figure 4: Visualization of the loss landscape of the initialization weights and optima obtained through GradOT and full model fine-tuning. The loss level of Emu. FT with GradOT is also highlighted.

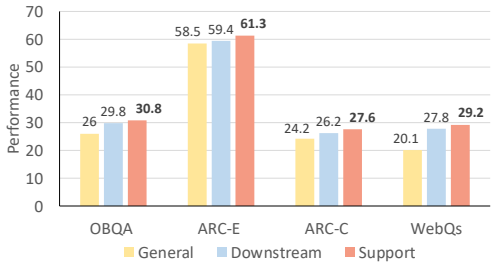


Figure 5: Comparison of different data types for statistical analysis in terms of plug-in results of our method.

tently outperform OT on the LLaMA-13B model, further demonstrating our method's effectiveness on larger models.

One significant advantage of GradOT is its efficiency. As further detailed in Appendix B.1, GradOT requires significantly less time to achieve results comparable to SoTA post-training methods.

4.3 Analytical Studies

This section presents key analytical studies while Appendix B provides additional analysis.

Impact of different components. The ablation study results, as presented in Table 4, provide valuable insights into the impact of the different components proposed on performance across the OBQA, ARC-C, and SciQ datasets. We selected OBQA and SciQ as they represent the most and least challenging multi-choice QA datasets among the candidates, allowing us to assess GradOT's effectiveness across different difficulty levels. Additionally, ARC-C, being the most challenging Closed-Book QA dataset, helps validate our proposed components for such tasks.

It is evident that the sole application of SCP improves plug-in performance but yields weak privacy protection, as indicated by the modest $\Delta = 1.3$. In contrast, using only DRD significantly enhances Δ compared to SCP alone, though it may lead to degradation in plug-in model performance. The combined use of both DRD and SCP achieves the highest compression ratio with the lowest r while also delivering the best Δ. Furthermore, this combination results in plug-in performance comparable to that of the non-compressed model, confirming the effectiveness of both the DRD and SCP components. Additionally, a comparison of the fourth-row results to the last two rows highlights the importance of the gradient-preserving compression score. Without this score, both plug-in performance and Δ experience a notable decline, underscoring the critical role of the proposed score in ensuring robust performance and privacy preservation. These findings collectively validate the efficacy of the individual components and the proposed score in achieving the desired balance between model compression, performance, and privacy protection.

Loss Landscape Analysis. Fig. 4 visualizes the loss landscape, illustrating that both the initialization weights and the optima obtained through GradOT and full model fine-tuning lie within the same basin and at the same level. This observation suggests that our gradient-preserving approach enables $\widehat{\mathcal{A}}'$ to converge to the same level as $\mathcal{A}'$, demonstrating the effectiveness of GradOT. Notably, the loss for Emu. FT with GradOT is positioned significantly further from both the fully fine-tuned and the GradOT Plug-in models, highlighting that GradOT sufficiently preserves privacy.

Impact of Data Type for Support Dataset. To protect the privacy of downstream data, we use a separate supporting dataset exclusively for statistical analysis to compute the first- and second-order derivatives, which are then used to calculate the Score. As shown in Fig. 5, we compare support datasets with general tasks using WikiText as the general dataset and the downstream dataset directly.

The results obtained from the downstream dataset are significantly better than those from the general dataset. This is likely due to the differing contributions of various weights for general and downstream tasks. The support dataset yields the best results, even surpassing the downstream data. We hypothesize that this may be due to the supporting dataset containing multiple datasets, offering better generalization for downstream tasks. In future work, we will explore how to select the support dataset to maximize generalizability.

5 Discussions

Effect of Gradient Gap Term. One may notice that previous studies (Kim et al., 2024; Osawa et al., 2023; Ren and Zhu, 2023) conduct importance estimation for model compression through the perspective of Fisher information, whose efficacy has been validated. Our methods also connect to these researches as we adapt Fisher information, but it is used to evaluate the gradient variations. Our results indicate that preserving the gradient of adapters based on Fisher information is feasible.

Indicator for Other Methods. Besides the effective proposed GradOT, our gradient-preserving compression score also serves as an indicator for general offsite-tuning-oriented compression methods, such as feature-based low-rank compression (Ji et al., 2024). Possible applications in broader cases can be explored in future work.

6 Conclusion

This paper demonstrates that the OT problem can be reformulated into a gradient-based form, from which the gradient-preserving compression score is derived. We introduce GradOT, a method that combines Dynamic Rank Decomposition and Selective Channel Pruning, both of which utilize the gradient-preserving compression score. Extensive experiments and analyses suggest that GradOT offers a promising solution for OT tasks.

Acknowledgements

This work was supported by Ant Group Postdoctoral Programme.

Limitation

One limitation of our approach is that the overall design of the proposed GradOT is based on the application to the transformer-based architecture, as described in Section 3.3. Therefore, GradOT may not be feasible for non-transformer-based models, e.g., Mamba. Notably, due to limited resources, we were unable to validate LLMs like the LLaMA3 70B. These larger models demonstrate enhanced language comprehension capabilities and, as a result, deliver superior performance.

References

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *AAAI*, volume 34, pages 7432–7439.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

Terence Jie Chua, Wenhan Yu, Jun Zhao, and Kwok-Yan Lam. 2023. Fedpeat: Convergence of federated learning, parameter-efficient fine tuning, and emulator assisted tuning for artificial intelligence foundation models with mobile edge computing. *arXiv preprint arXiv:2310.17491*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Association for Computational Linguistics*, pages 2924–2936.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig. 2021. Laplace redux-effortless bayesian deep learning. *Advances in Neural Information Processing Systems*, 34:20089–20103.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Baobao Chang, Xu Sun, and Zhifang Sui. 2024. A survey on in-context learning. In *Conference on Empirical Methods in Natural Language Processing*, pages 1107–1128.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. GLM: general language model pretraining with autoregressive blank infilling. In *Association for Computational Linguistics*, pages 320–335.

Carl Eckart and Gale Young. 1936. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Junyuan Hong, Jiachen T. Wang, Chenhui Zhang, Zhangheng LI, Bo Li, and Zhangyang Wang. 2024. Dp-opt: Make large language model your privacy-preserving prompt engineer. In *International Conference on Learning Representations*.

Yixin Ji, Yang Xiang, Juntao Li, Wei Chen, Zhongyi Liu, Kehai Chen, and Min Zhang. 2024. Feature-based low-rank compression of large language models via bayesian optimization. *In Findings of the Annual Meeting of the Association for Computational Linguistics*.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Association for Computational Linguistics*.

Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W. Mahoney, and Kurt Keutzer. 2024. Squeezellm: Dense-and-sparse quantization. In *International Conference on Machine Learning*.

Frederik Kunstner, Philipp Hennig, and Lukas Balles. 2019. Limitations of the empirical fisher approximation for natural gradient descent. *Advances in neural information processing systems*, 32.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations. In *Conference on Empirical Methods in Natural Language Processing*, pages 785–794.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: adversarial attack against BERT using BERT. In *Conference on Empirical Methods in Natural Language Processing*, pages 6193–6202.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2024a. Visual instruction tuning. *Advances in Neural Information Processing Systems*, 36.

Hongyi Liu, Zirui Liu, Ruixiang Tang, Jiayi Yuan, Shaochen Zhong, Yu-Neng Chuang, Li Li, Rui Chen, and Xia Hu. 2024b. Lora-as-an-attack! piercing llm safety under the share-and-play scenario. *arXiv preprint arXiv:2403.00108*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Conference on Empirical Methods in Natural Language Processing*.

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

Kazuki Osawa, Shigang Li, and Torsten Hoefler. 2023. Pipefisher: Efficient training of large language models using pipelining and fisher information matrices. *Proceedings of Machine Learning and Systems*, 5:708–727.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

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

Adityanarayanan Radhakrishnan, Daniel Beaglehole, Parthe Pandit, and Mikhail Belkin. 2024. Mechanism for feature learning in neural networks and backpropagation-free machine learning models. *Science*, 383(6690):1461–1467.

Siyu Ren and Kenny Q Zhu. 2023. Low-rank prune-and-factorize for language model compression. *arXiv preprint arXiv:2306.14152*.

Hippolyt Ritter, Aleksandar Botev, and David Barber. 2018. A scalable laplace approximation for neural networks. In *6th international conference on learning representations, ICLR 2018-conference track proceedings*, volume 6. International Conference on Representation Learning.

Hassan Sajjad, Fahim Dalvi, Nadir Durrani, and Preslav Nakov. 2023. On the effect of dropping layers of pre-trained transformer models. *Computer Speech & Language*, 77:101429.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Tanmay Singh, Harshvardhan Aditya, Vijay K. Madisetti, and Arshdeep Bahga. 2024. Whispered tuning: Data privacy preservation in fine-tuning llms through differential privacy. *Journal of Software Engineering and Applications*, 17(1):1–22.

Zhaorui Tan, Xi Yang, Qiufeng Wang, Anh Nguyen, and Kaizhu Huang. 2024. Interpret your decision: Logical reasoning regularization for generalization

in visual classification. In *Advances in Neural Information Processing Systems*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30.

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

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. In *EMNLP Workshop*, pages 94–106. Association for Computational Linguistics.

Noam Wies, Yoav Levine, and Amnon Shashua. 2024. The learnability of in-context learning. *Advances in Neural Information Processing Systems*, 36.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, and Hongseok Namkoong. 2022. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7959–7971.

Guangxuan Xiao, Ji Lin, and Song Han. 2023. Offsite-tuning: Transfer learning without full model. *arXiv preprint arXiv:2302.04870*.

Adam X Yang, Maxime Robeyns, Xi Wang, and Laurence Aitchison. 2024. Bayesian low-rank adaptation for large language models. In *international conference on learning representations*.

Kai Yao, Penglei Gao, Lichun Li, Yuan Zhao, Xiaofeng Wang, Wei Wang, and Jianke Zhu. 2024. Layer-wise importance matters: Less memory for better performance in parameter-efficient fine-tuning of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1977–1992.

Kai Yao, Zhaorui Tan, Tiandi Ye, Lichun Li, Yuan Zhao, Wenyan Liu, Wei Wang, and Jianke Zhu. 2025. Scaleot: Privacy-utility-scalable offsite-tuning with dynamic layerreplace and selective rank compression. In *AAAI*.

Tiandi Ye, Cen Chen, Yinggui Wang, Xiang Li, and Ming Gao. 2024. Bapfl: You can backdoor personalized federated learning. *Transactions on Knowledge Discovery from Data*, 18(7):166.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Association for Computational Linguistics*, pages 4791–4800.

Kaiyan Zhang, Ning Ding, Biqing Qi, Xuekai Zhu, Xinwei Long, and Bowen Zhou. 2023a. CRaSh: Clustering, removing, and sharing enhance fine-tuning without full large language model. In *Conference on Empirical Methods in Natural Language Processing*, pages 9612–9637.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2023b. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. 2022. Conditional prompt learning for vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16816–16825.

Andy Zou, Zifan Wang, J Zico Kolter, and Matt Fredrikson. 2023. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*.

A More Details of Our Method

A.1 Total Differential Details

Given a function $f(x_1, x_2, \ldots, x_n)$ with variables $x_1, x_2, \ldots, x_n$, its total differential df can be expressed as:

$$df = \sum_{i=1}^{n} \frac{\partial f}{\partial x_i} dx_i, |dx_i| < \epsilon, \tag{14}$$

where ϵ is a small value. By considering the sample as a constant and treating w as the input x, with $f = \ell(w)$ representing the loss calculation, we can derive the total differential of $\ell(w)$ as follows:

$$\partial \ell(w) = \sum_{i=n_1}^{n_2} \frac{\partial \ell}{\partial w_i} \odot dw_i. \tag{15}$$

dw_i can be considered as the δ_i between w_i and $\hat{w}_i$. Therefore, we have:

$$\ell(\hat{w}) - \ell(w) = \sum_{i=n_1}^{n_2} \frac{\partial \ell}{\partial w_i} \odot \delta_i, \tag{16}$$

where $\odot$ is the inner product.

A.2 Efficient Compression

To reduce time and computational cost during model compression, we calculate and rank the scores of each weight component in ascending order, then select the top K components to satisfy the compression rate requirement. Specifically, in the Dynamic Rank Decomposition (DRD) method, a given weight component $w_{(k)}$ is expressed as the product of the k^{th} eigenvalue and its corresponding eigenvector:

$$w_k = \boldsymbol{U}_{:,k} \boldsymbol{\Sigma}_{k,k} \boldsymbol{V}^T_{k,:}. \tag{17}$$

Conversely, the Selective Channel Pruning (SCP) method represents the weight of the k^{th} channel:

$$w_k = \{\boldsymbol{W}^{up}_{:,k}, \boldsymbol{W}^{down}_{k,:}\} \tag{18}$$

We then compute the scores of these weight components $\mathrm{GCS}(w_{(k)})$ and, upon completion, select the components with the highest K scores to aggregate them $\delta_i = \sum w_{(k)}$, where the selected index set is $s = \{k|\mathrm{Score}(w_{(k)}) < \mathrm{Score}(w_{(K)})\}$. This approach enables us to calculate the weight scores once, selecting those corresponding to the top K scores in both DRD and SCP methods, thereby achieving compression. It is important to acknowledge that this method may introduce some errors during score calculation, as $||\frac{\partial^2 \ell}{\partial w_i^2} \delta_i||_1 \leq \sum ||\frac{\partial^2 \ell}{\partial w_i^2} w_{(k)}||_1$; however, it substantially enhances computational efficiency and eliminates the need for search techniques. This strategy significantly enhances the efficiency of the model compression.

A.3 Implementation Details

For downstream training, we employ the AdamW Optimizer with a grid search of learning rates. We perform a learning rate tuning process on a grid of values and report the runs with the highest emulator performance, where {2e-5, 5e-5, 1e-4, 2e-4, 3e-4} for OPT-1.3B, OPT-6.7B, and LLaMA-7B. Due to the large range differences in our scoring calculations, we roughly searched for λ within {1e1, 1e2, 1e3, 1e4, 1e5}, which implies that better parameters might exist if a more refined search is conducted. All experiments are conducted on a workstation with one A100 80G.

A.4 Dataset Statistics

For downstream tasks, the detailed dataset statistics can be referred to Table 5. Meanwhile, we showcase the instructions formats of different datasets in Tables 9 and 10.

Dataset	# Train	# Test	Answer
Downstream Tasks			
OBQA	5.0K	500	Option
PIQA	16.1K	1,830	Option
ARC-E	1.1K	2,376	Option
ARC-C	2.3K	1,172	Option
HellaSwag	39.9K	10,042	Option
SciQ	11.7k	1,000	Option
WebQs	3.8k	2,032	Option
RACE	62.4k	3,498	Option

Table 5: The statistics of datasets for evaluation. # Train and # Test denote the number of training and test samples, respectively.

B Additional Experiments and Analysis

B.1 Consumption of Compression Cost.

We present the time consumption associated with our proposed GradOT. It is necessary to precompute the statistical information of the model on the support dataset. Although this process may be time-consuming, it worth noting that it only needs to be completed once for each model. Despite the significant memory demands inherent in gradient

computation, our training-free approach allows for linear-by-linear computation, facilitating scalability to large models. We report our results using a support dataset size of 1500 on a single 80G A100 GPU. Due to the memory limitation, for 7B models, we divided the emulator into 7 groups, with each group containing 4 transformer blocks for separate computation. As shown in Tables 6 and 7, gradient calculation takes 8 minutes for OPT-1.3B and 97 minutes for LLaMA-7B. In contrast, model compression is considerably faster than the preparation stage, taking only 1 minute for OPT-1.3B and 8 minutes for LLaMA-7B. Compared with post-training-based methods like OT and ScaleOT, as shown in Table 7, which require tons of computation and time cost (e.g., OT and ScaleOT cost 60 and 5 hours on 1.5B models, respectively.), our method significantly reduces the time and computational costs involved in model compression.

Models	Grad. Calculation	Model Compression
OPT-1.3B	8 mins	1 mins
LLaMA-7B	97 mins	8 mins

Table 6: Time consumption of conducting GradOT on different LLMs.

Methods	**OPT-1.3B**	**LLaMA-7B**
Post-training Compression		
OT	60 hours	-
ScaleOT	5 hours	57 hours
Training-free Compression		
GradOT	10 mins	96 mins

Table 7: Time consumption comparison on OPT-1.3B and LLaMA-7B.

B.2 Sensitive Study

λ_{mha}	OBQA	
	Emu. FT	Plug-in
0	29.2	31.6
1e3	28.6	31.2
1e4	27.0	30.8
1e5	26.0	30.2

λ_{mlp}	OBQA	
	Emu. FT	Plug-in
0	29.6	33.0
1e1	29.2	30.8
1e2	27.0	30.8
1e3	24.6	26.4

Table 8: Sensitive study of λ for OPT-1.3B. Default parameters are marked in **bold**.

The sensitivity study of λ_{mha} and λ_{mlp} in Table 8 examines their impact on the performance of the Emulator Fine-Tuned (Emu. FT) and Plug-in models on the OBQA dataset. It can be noticed that the Plug-in performance is not very sensitive to λ_{mha}. As λ_{mha} increases, the Emu. FT performance consistently declines from 29.2 to 26.0, while the Plug-in model maintains relatively stable performance, decreasing only slightly from 31.6 to 30.2. This suggests that gradient-preserving compression score on multi-head attention (MHA) negatively affects the Emu. FT model and improving the model's privacy protection, the Plug-in model remains robust.

Similarly, increasing λ_{mlp} results in a decline in Emu. FT performance drops from 29.6 to 27.0, while the Plug-in model also experiences a slight performance drop from 33.0 to 30.8 while $\lambda_{mlp} \leq 1e3$. Further increasing λ_{mlp} to $1e3$ leads to a significantly declined performance for both Emu. FT and Plug-in.

Notably, the Plug-in model consistently outperforms Emu. FT across all settings, reinforcing the effectiveness of the proposed approach in privacy protection and preserving Plug-in performance. The default hyperparameter settings for OPT-1.3B, **1e4** for λ_{mha} and **1e2** for λ_{mlp}, strike a balance between Plug-in model performance and privacy protection.

B.3 Compression Ratio for Emulator.

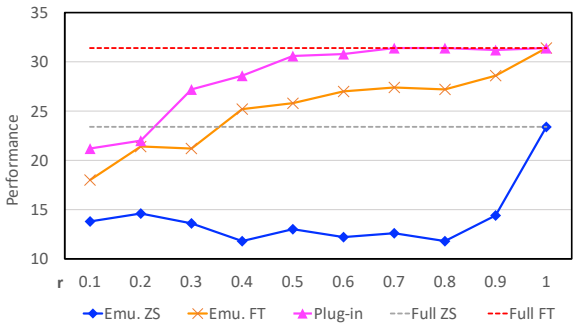


Figure 6: Performance of applying GradOT to OPT-1.3B with increased r on OBQA dataset.

Fig. 6 visualizes the results of applying GradOT to the OPT-1.3B model with varying values of r. The constant gap between the plug-in results and the emulator's fine-tuned results for $r < 1$ suggests that GradOT can consistently preserve model privacy. Notably, when $r \geq 0.5$, the plug-in model not only maintains privacy but also achieves performance comparable to, or even exceeding, that of the fully fine-tuned model. This highlights the effectiveness of GradOT in preserving privacy without sacrificing performance, making it a promising

approach for OT with various compression ratios.

B.4 Score Term (1) Approximates the Loss Discrepancy Between the Original and Noised Models.

To evaluate the effectiveness of Score Term (1), we add random noise at varying scales into the middle-layer weights of the OPT-1.3B model and measure the error between the estimated values from Score Term (1) and the actual losses. Specifically, we progressively increase the noise magnitude, scaling its Frobenius norm from 0.001 to 1. As shown in Fig. 7, despite the increasing noise, the estimation error of Score Term (1) remains stable within ± 0.02, demonstrating its robustness. These results validate the accuracy of Score Term (1) in approximating loss discrepancies and, consequently, support the efficacy of Score Term (2).

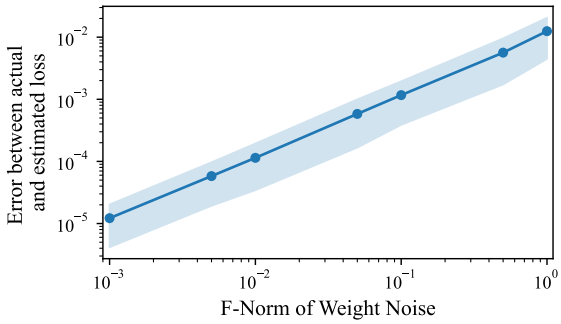


Figure 7: Error between the actual loss and estimated loss with Eq. (15) under different weight noise. Both the horizontal and vertical axes are scaled by log.

OpenBookQA

Poison causes harm to which of the following?
a Tree

As a car approaches you in the night
the headlights become more intense

PIQA

Question: To create a makeshift ice pack,
Answer:
take a sponge and soak it in water. Put the sponge in a refrigerator and let it freeze. Once frozen, take it out and put it in a ziploc bag. You can now use it as an ice pack.

Question: What should I use as a stain on a wooden bowl I've just made.
Answer:
You should coat the wooden bowl with a butcher block oil & finish per manufacturer directions.

SciQ

One type of radioactivity is alpha emission. What is an alpha particle? What happens to an alpha particle after it is emitted from an unstable nucleus?.
Question: Alpha emission is a type of what?
Answer:
radioactivity

All radioactive decay is dangerous to living things, but alpha decay is the least dangerous.
Question: What is the least dangerous radioactive decay?
Answer:
alpha decay

RACE

Article: Understanding the process of making career choices and managing your career is a basic life skill that everyone should understand.
Your career decisions have such a profound effect on all aspects of your life. It's important to have the knowledge and resources needed to make smart, informed decisions. Whether you are looking for a new job, aiming to take the next step at your current job or planning your retirement options, you are making career decisions. Using good resources and the guidance of a career counselor can help you to make those decisions well.
Many people mistakenly believe that choosing a career is a one-time event that happens some time in early adulthood. However, career management is actually a life-long process, and we continue to make consequential career choices over the years. When people want to take action in their career, career management and job search are about so much more than writing a good resume. If you learn about and act on the following areas of career management, you'll be rewarded throughout your career.
Your interests, abilities, values, personal needs and realities should all be taken into account in any career decision making process. You spend countless hours at work, and it impacts your life in so many ways; it makes sense that you should be fully informed before making such profound decisions.
Do you know how many different career choices are available to you? Both The Dictionary of Occupational Titles (American) and The National Occupational Classification (Canadian) list well over 20,000 different job titles. So unless you've actively explored a variety of career options, there's a very good chance that there are great possibilities available to you, and you don't even realize they exist.
Match your understanding of yourself with your understanding of possible career options. Once you have developed a good understanding of yourself, you will be able to combine that self-knowledge with your career and labor market research to determine potential careers that are a great fit for you.
When you've made a well informed decision, then you're ready to make it happen. Making use of good career guidance and resources will help you to acquire the education, skills, and experience needed to get the job and learn and implement effective job search strategies.
Time spent understanding your needs, researching your career options and developing outstanding job search skills, guided by great career resources, is a powerful investment in your future.

Question: It can be inferred that _ .

Answer:
career decision is misunderstood by many people because they don't take it as a life-long proces.

Table 9: Instructions format of Multi-Choice QA task.

ARC-E
Question: A chewable calcium carbonate tablet is a common treatment for stomach discomfort. Calcium carbonate is most likely used as this type of medicine because calcium carbonate Answer: *neutralizes digestive acid.*
Question: Which two body systems are directly involved in movement? Answer: *muscular and skeletal*
ARC-C
Question: A fold observed in layers of sedimentary rock most likely resulted from the Answer: *converging of crustal plates.*
Question: As part of an experiment, an astronaut takes a scale to the Moon and weighs himself. The scale reads 31 pounds. If the astronaut has a mass of about 84 kilograms, which are the approximate weight and mass of the astronaut when standing on the Earth? Answer: *186 pounds and 84 kilograms*
WebQS
Question: where is shoreview mn? Answer: *Ramsey County*
Question: what is the currency in the dominican republic called? Answer: *Dominican peso*
HellaSwag
Getting a haircut: He then combs it and blow dries his hair after styling it with gel. He uses an electric clipper to groom the sideburns and the temples. He *also trims the back and sides of his head with the clippers.*
Getting a haircut: The man in the center is demonstrating a hairstyle on the person wearing the blue shirt. The man in the blue shirt *sits on the chair next to the sink.*

Table 10: Instructions format of Closed-Book QA and Sentece Completion task.