Breaking the Frozen Subspace: Importance Sampling for Low-Rank Optimization in LLM Pretraining

Haochen Zhang[1], Junze Yin[1], Guanchu Wang[2], Zirui Liu[3], Lin F. Yang[4], Tianyi Zhang[1], Anshumali Shrivastava[1], Vladimir Braverman[1,5]

[1]Rice University, [2]University of North Carolina at Charlotte, [3]University of Minnesota Twin Cities, [4]University of California, Los Angeles, [5]Johns Hopkins University

Abstract

Low-rank optimization has emerged as a promising approach to enabling memory-efficient training of large language models (LLMs). Existing low-rank optimization methods typically project gradients onto a low-rank subspace, reducing the memory cost of storing optimizer states. A key challenge in these methods is selecting suitable subspaces to ensure an effective optimization trajectory. Most existing approaches select the dominant subspace to preserve gradient information, as this intuitively provides the best approximation. However, we find that in practice, the dominant subspace stops changing during pretraining, thereby constraining weight updates to similar subspaces. In this paper, we propose importance sampling for low-rank optimization in LLM pretraining with a provable convergence guarantee, which the dominant subspace approach does not have. Empirically, we demonstrate that our method significantly outperforms previous methods in LLM pretraining tasks.

1 Introduction

Large language models (LLMs), pretrained on next-token prediction tasks, achieve human-level text generation capabilities and exhibit zero-shot transferability to various downstream tasks [3]. They are also fine-tuned or aligned with human preferences to be expert in downstream tasks [37, 30]. Over the past few years, there has been rapid progress in LLM development, characterized by consistent growth in the number of trainable parameters and the scale of datasets [2, 17, 8, 1]. The parameter count in language models has increased from 100 million [33] to over a hundred billion [5]. However, despite their enhanced expressiveness, such large models demand extensive GPU memory for pretraining [29]. Thus, a critical question arises:

How can we improve the memory efficiency of LLM pretraining?

In LLM pretraining, Adam is commonly used as the optimizer due to its superior optimization performance. However, a key limitation of Adam is its memory requirement, as it necessitates storing two optimizer states, each consuming as much memory as the model itself. This poses a significant challenge, given the substantial memory demands of the model's parameters. To address this issue, researchers have explored low-rank optimization, where gradients are projected onto a low-rank subspace to reduce the memory consumption of optimizer states. These states are then projected back to their original size when updating the weights. For example, GaLore [42] and Q-GaLore [41] project gradients onto subspaces defined by the leading singular vectors corresponding to the largest singular values, a technique referred to as the dominant subspace. FLora [11] and GoLore [12], on the other hand, utilize unbiased random low-rank projections for gradients, employing the Johnson–Lindenstrauss transform. Grass [28] introduces sparse low-rank projections, which further

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

reduce the gradient memory footprint as well as the computation and communication costs compared to dense low-rank projections. Lastly, Fira [4] builds on GaLore by fully leveraging the error in gradient low-rank approximation to achieve improved performance.

These methods are powerful because: 1) the gradients of LLMs during pretraining exhibit an intrinsic low-rank structure, making them well-suited for compression using low-rank approximation, and 2) low-rank approximation can be applied not only to Adam but also to other optimizers that use state information. For instance, Adafactor [35] employs rank-1 factorization on the second moment in Adam to reduce the memory required for storing the second moment. Adam-mini [40] eliminates over 99% of the effective learning rate in the second moment of Adam while achieving performance on par with—or even better than—Adam. Additionally, [7] and [21] propose low-precision optimizers with 8-bit and 4-bit optimizer states. Low-rank optimization integrates seamlessly with these Adam variants, further highlighting its importance and underscoring why it deserves significant attention.

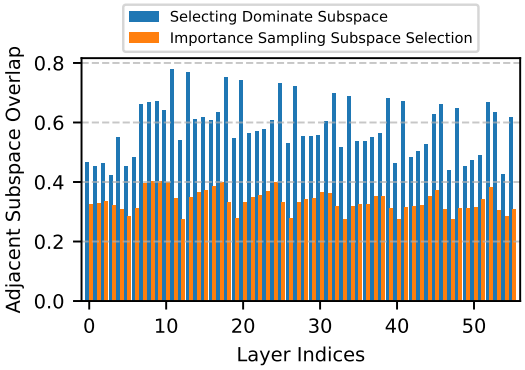


Figure 1: Adjacent subspace overlap of low-rank optimizer using difference subspace selection methods. Our importance sampling subspace selection can lower the overlap between adjacent subspaces, thus enabling better exploration of more different subspaces in the optimization trajectory.

A central question in low-rank optimization is how to maintain the performance of pretrained LLMs while using memory-efficient optimizers, as compared to full-rank optimization. One common paradigm in existing low-rank optimization methods is to update weights within the dominant subspace for a certain number of iterations and periodically update this dominant subspace. Nonetheless, the dominant subspaces of gradients in many layers stabilize almost completely after the early stages of pretraining [41]. Consequently, the weight updates during different periods predominantly remain within the same low-rank subspace, resulting in cumulative weight updates that struggle to achieve high rank. This limitation significantly hampers the language modeling capabilities of pretrained LLMs. Thus, it is natural to ask:

Is it possible to overcome the low-rank bottleneck of existing low-rank optimization methods with minimal additional overhead?

In this paper, we provide a positive answer to this question. We propose a novel method for subspace selection in low-rank optimization by introducing an appropriate degree of randomness in the selection process. In summary, the contributions of this study are as follows:

- We observe that highly similar adjacent subspaces in existing low-rank optimization methods diminish the diversity of weight updates, degrading the performance of pretrained LLMs.
- To address frozen dominant subspace phenomenon and the low-rank bottleneck of update in existing low-rank optimization methods, we propose an Importance **SA**mpling method for Low-**RA**nk optimization (**SARA**). This method enables low-rank optimizers to explore a broader range of subspaces in the optimization trajectory. Specifically, the low-rank subspace is spanned by r singular vectors sampled from m singular vectors for a gradient $G \in \mathbb{R}^{m \times n}$. Figure 1 illustrates how SARA reduces the overlap between adjacent subspaces during LLM pretraining.
- SARA can be integrated with various low-rank optimization methods, such as GaLore and Fira. It is robust to second-moment factorization and low-precision optimizer state storage. On pretraining tasks for the LLaMA model at different sizes, SARA consistently outperforms dominant subspace selection and reduces the performance gap between low-rank optimizers and full-rank Adam by up to 46.05%.
- From a theoretical aspect, We prove that SARA achieves a comparable convergence rate as GoLore [12] (Theorem 3.4, proof details are deferred to Appendix A) whereas delivering better empirical results (Section 4 and Appendix E).

Roadmap. In Section 2, we present the update rules of GaLore-Adam and Fira-Adam. In Section 3, we describe our methodology for using importance sampling to improve the algorithmic design of low rank optimizers—GaLore and Fira—and provide the convergence guarantee of SARA. In Section 4, we present experimental results showing that SARA consistently outperforms dominant subspace selection. In Section 5, we discuss related work. Finally, in Section 6, we conclude the paper.

2 Preliminaries

In this section, we present the background required for our theoretical analysis and experiments. In our experiments (Section 4), we apply SARA to two low-rank optimization methods, GaLore and Fira, both of which can be combined with stateful optimizers (e.g., Adam, Adafactor, and Adam-mini).

To ensure clarity, the update rules for GaLore-Adam and Fira-Adam are briefly explained here. For more detailed explanations, please refer to the original papers [42, 4]. In presenting these methods, we show the update rules for the weights of a single layer in the neural network. We assume that the gradient at the t-th iteration is a matrix $G^{(t)} \in \mathbb{R}^{m\times n}$. Without loss of generality, we assume that $m < n$ and use r to represent the rank of the low-rank subspace.

Update Rules of GaLore-Adam GaLore-Adam [42] requires storing an orthogonal matrix $P^{(t)} \in \mathbb{R}^{m\times r}$ that satisfies $(P^{(t)})^\top P^{(t)} = I_r$, which is updated periodically. Similar to full-rank Adam, GaLore-Adam also stores the first moment $M^{(t)} \in \mathbb{R}^{r\times n}$ and the second moment $V^{(t)} \in \mathbb{R}^{r\times n}$ for each layer's weights, and updates the weights $W^{(t)}$ as follows: $R^{(t)} = (P^{(t)})^\top G^{(t)}$, $M^{(t)} = \beta_1 M^{(t-1)} + (1-\beta_1)R^{(t)}$, $V^{(t)} = \beta_2 V^{(t-1)} + (1-\beta_2)R^{(t)} \circ R^{(t)}$, $N^{(t)} = \alpha P^{(t)} \frac{M^{(t)}}{\sqrt{V^{(t)}}+\xi}$, and $x^{(t)} = x^{(t-1)} - \eta \cdot N^{(t)}$. Here, β_1 and β_2 are hyperparameters for the online updates of $M^{(t)}$ and $V^{(t)}$, as in Adam. The parameter η denotes the learning rate, and ξ is a small positive constant for numerical stability.

Update Rules of Fira-Adam Similar to GaLore-Adam, Fira also needs to store $M^{(t)}$, $V^{(t)}$, and $P^{(t)}$. The difference is that Fira-Adam additionally utilizes the low-rank approximation residual to update $W_l^{(t)}$. Let $S^{(t)} = (I - P^{(t)}(P^{(t)})^T)G^{(t)}$ and $x^{(t)} = x^{(t-1)} - \eta \cdot N^{(t)} - \eta \cdot \phi(S^{(t)})$. where $S^{(t)}$ represents the low-rank approximation error, $\phi(\cdot)$ represents a scaling function in Fira [4], and $N^{(t)} = \alpha P^{(t)} \frac{M^{(t)}}{\sqrt{V^{(t)}}+\xi}$ is calculated same as GaLore-Adam above.

3 Methodology

In this section, we first illustrate the adverse effects of a frozen dominant subspace in mini-batch gradients (Section 3.1). To address this issue, we then propose SARA for low-rank optimization (Section 3.2). Finally, we present a convergence analysis of low-rank optimization using SARA (Section 3.3).

3.1 Frozen Dominant Subspace of Mini-batch Gradient

[41] observes that the cosine similarity between adjacent dominant subspaces approaches 1.0 in some layers after a certain stage of LLM pretraining, indicating that the dominant subspace of the gradient almost stops evolving. We observe a similar phenomenon in our experiment as well. Figure 2 shows the average result of dominant subspace overlap in different layers across all blocks at different iterations. We notice that dominant subspace overlaps are low in all layers at the early stage of pretraining, but they increase drastically as pretraining progresses, eventually becoming stable at different levels. Among all layers, gate_proj and up_proj exhibit the highest subspace overlaps. Intuitively, a high overlap between adjacent subspaces is harmful for low-rank optimization. Considering an extreme case, when the overlap reaches 1.0, the low-rank optimizer can only change the weights within a fixed low-rank subspace. However, when the low-rank subspace shifts significantly over time, the overall weight update—formed by summing updates from various low-rank subspaces—can overcome the constraints of the low-rank bottleneck. For readability, we refer to this phenomenon as the frozen dominant subspace.

Algorithm 1 Low-rank Optimization with SARA

Input: The l-th layer weight $x_l^{(t)} \in \mathbb{R}^{m_l \times n_l}$, for all $l \in [N]$. Learning rate η, scale factor α, decay rates β_1, β_2, rank r, subspace change frequency $\tau \in \mathbb{Z}_+$, small constant for numerical stability ξ.
Initialize: for all $l \in [N]$ $V_l^{(0)}, M_l^{(0)} \in \mathbb{R}^{r \times n_l} \leftarrow 0$
for $t = 1 \to T$ **do**
    for $l = 1 \to N$ **do**
        Compute the mini-batch gradient: $G_l^{(t)} \in \mathbb{R}^{m_l \times n_l}$
        $P_l^{(t)} \leftarrow \text{SARA}(G_l^{(t)}, \tau)$ ▷ see Algorithm 2
        $\mathcal{S} \leftarrow \{V_l^{(t-1)}, M_l^{(t-1)}, x_l^{(t)}, P_l^{(t)}, G_l^{(t)}, \beta_1, \beta_2, \xi, \eta, \alpha\}$ ▷ input parameters
        $x_l^{(t)} \leftarrow \text{GaLore-Adam}(\mathcal{S})$ or $\text{Fira-Adam}(\mathcal{S})$ ▷ see Section 2.
    end for
end for
Return $x^{(T)} = (x_1^{(T)}, x_2^{(T)}, \cdots, x_N^{(T)})$

3.2 SARA: Importance SAmpling for Low-RAnk Optimization

Algorithm 2 SARA: Importance sampling subspace selection for low-rank optimization

Input: The mini-batch gradient at the iteration t, $G_l^{(t)} \in \mathbb{R}^{m \times n_l}$, where $l \in [N]$ denotes the layer. Subspace change frequency $\tau \in \mathbb{Z}_+$.
if $t \bmod \tau = 0$ **then**
    $U_l^{(t)}, S_l^{(t)}, V_l^{(t)} \leftarrow \text{SVD}(G_l^{(t)})$
    $\mathcal{I} \leftarrow \text{Sample}([m], \text{num} = r, \text{weight} = S_l^{(t)})$
    $\mathcal{I} \leftarrow \text{Sort}(\mathcal{I})$
    $P_l^{(t)} \leftarrow U_l^{(t)}[:, \mathcal{I}]$
else
    $P_l^{(t)} \leftarrow P_l^{(t-1)}$ ▷ Reuse the previous projector
end if
Return $P_l^{(t)}$

To overcome the problem of the frozen dominant subspace problem, we propose SARA to construct low-rank subspace. Low-rank optimization with SARA is given in Algorithm 1. It can be seen that SARA does not change the overall structure of the original low-rank optimization algorithm but is a plug-and-play substitute for dominant subspace selection. Algorithm 2 gives the procedure of SARA. Line 4 denotes the weighted sampling without replacement. More precisely, each of the m left singular vectors is equipped with a weight $\omega_i \in (0, 1)$ proportional to its corresponding singular value S_i,

$$\omega_i = \frac{S_i}{\sum_{j=1}^{m} S_j}.$$

For an index set sample $\mathcal{I} = (I_1, \cdots, I_r)$, the sampling probability

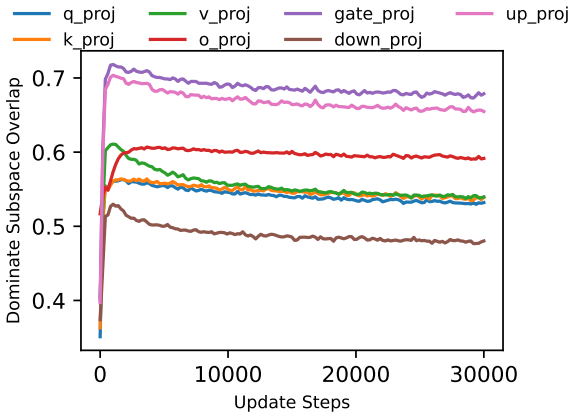


Figure 2: The average mini-batch gradient dominant subspace overlap in different linear layers over 8 blocks in the LLaMA-60M model during pretraining. We measure the overlap between adjacent subspaces every 200 iterations.

can be written as

$$\mathbb{P}\{(I_1, \cdots, I_r) = (i_1, \cdots, i_r)\} = \prod_{k=1}^{r} \frac{\omega_{i_k}}{1 - \omega_{i_1} - \cdots - \omega_{i_{k-1}}}$$

Line 5 sorts the sampled indices in ascending order so that the newly updated subspace basis vectors can align with optimizer states well. Line 6 constructs the orthogonal basis of the new subspace.

By using weighted sampling without replacement, we make adjacent subspaces more different and make the optimization trajectory not be trapped in too similar subspaces during training. Another advantage of SARA is that it brings negligible extra overhead—for instance, computing an SVD on a 2048×2048 matrix takes 0.34 seconds, while sampling adds just 0.0005 seconds on average.

3.3 Provable Convergence of SARA

[12] points out that selecting the dominant subspace in low-rank optimization, as done in GaLore, does not always guarantee convergence to the optimal solution. Although GoLore ensures provable convergence, it does not significantly close the performance gap between GaLore-Adam and full-rank Adam in pretraining tasks, as reported in [12]. In this section, we show that SARA achieves provable convergence, representing a key advantage over GaLore and comparable to GoLore. The empirical results of SARA are shown in Section 4 and Appendix E, representing a key advantage over GaLore and GoLore.

We treat an LLM as a neural network with N layers, and each layer has a weight matrix, i.e., for all $l \in [N]$, $x_l \in \mathbb{R}^{m\times n_l}$. Without loss of generality, we assume that $m \leq n_l$. In practice, most LLMs do not have biases for attention blocks and MLP blocks, and low-rank optimization is only applied to the weight matrix, but not to biases. Therefore, this abstraction is reasonable. Mathematically, our objective function is $f : \mathbb{R}^{m\times n_1} \times \mathbb{R}^{m\times n_2} \times \cdots \times \mathbb{R}^{m\times n_N} \to \mathbb{R}$. For all $x = (x_1, \ldots, x_l, \ldots, x_N), y = (y_1, \ldots, y_l, \ldots, y_N) \in \mathrm{dom}(f)$, we denote $\nabla_l f(x)$ and $\nabla_l f(y)$ as $\frac{\partial f}{\partial x_l} \in \mathbb{R}^{m\times n_l}$ and $\frac{\partial f}{\partial y_l} \in \mathbb{R}^{m\times n_l}$, respectively. Below, we use the following two assumptions similar to [12].

Assumption 3.1 (L-smoothness)**.** Let $f : \mathbb{R}^{m\times n_1} \times \mathbb{R}^{m\times n_2} \times \cdots \times \mathbb{R}^{m\times n_N} \to \mathbb{R}$ be our objective function. Let $L > 0$. For all $l \in [N]$, we let $x = (x_1, \ldots, x_l, \ldots, x_N), y = (y_1, \ldots, y_l, \ldots, y_N) \in \mathrm{dom}(f)$ be any arbitrary N-tuples satisfying that if $i \in [N] \setminus \{l\}$, then $x_i = y_i$. We assume f is L-smooth that it satisfies:

$$\|\nabla_l f(x) - \nabla_l f(y)\|_F \leq L \|x_l - y_l\|_F .$$

Assumption 3.2 (Bounded, Centered, and Independent Mini-batch Gradient Noise)**.** Let $\nabla_l f(x^{(t)}) \in \mathbb{R}^{m\times n_l}$ be the gradient of our objective function for the l-th layer at the t-th iteration, where $t \in \mathbb{Z}_+$. Let $G_l^{(t)} \in \mathbb{R}^{m\times n_l}$ be the mini-batch gradient which is the noisy version of $\nabla_l f(x^{(t)})$. For all $l \in [N]$, we assume there exists a least upper bound $\sigma_l^2 \in \mathbb{R}$ for $\|G_l^{(t)} - \nabla_l f(x^{(t)})\|_F^2$, namely

$$\left\| G_l^{(t)} - \nabla_l f(x^{(t)}) \right\|_F^2 \leq \sigma_l^2$$

and

$$\mathbb{E}\left[G_l^{(t)}\right] = \nabla_l f(x^{(t)}).$$

Furthermore, we define $\sigma^2 := \sum_{l=1}^{N} \sigma_l^2$.

To analyze the convergence of SARA, we characterize the error introduced by projecting gradients onto the sampled low-rank subspaces. Specifically, we bound the discrepancy between the original gradient and its projection under the importance sampling scheme of SARA. This projection error plays a central role in the convergence analysis, as it quantifies how well the sampled subspace preserves gradient information. Because of the page limit, we defer its proof to Appendix A.

Lemma 3.3 (Error of SARA's Projection, see Lemma A.2 for proof)**.** *Let τ be the update period of SARA, and r be the rank of low-rank subspace in SARA. For all $i \in [m], l \in [N], k \in \mathbb{N}$, let $p_l^{(t)}(i)$ denote the probability that the i-th basis vector is selected for the l-th layer at time $t = k\tau$, and*

define $\delta_l^{(t)} := \min_{i\in[m]} p_l^{(t)}(i)$, $\delta := \min_{l\in[m],t\geq 0} \delta_l^{(t)}$. *Let* $P_l^{(t)} \in \mathbb{R}^{m\times r}$ *denote the orthonormal projection matrix and let* $\nabla_l f(x^{(t)}) \in \mathbb{R}^{m\times n_l}$ *be the gradient matrix of the* l*-th layer at time* t*. Then, the following inequality holds:*

$$\mathbb{E}\left[\left\|\left(I - P_l^{(t)}(P_l^{(t)})^\top\right)\nabla_l f(x^{(t)})\right\|_F^2\right] \leq (1-\delta)\cdot\mathbb{E}\left[\left\|\nabla_l f(x^{(t)})\right\|_F^2\right].$$

With the projection error bounded in Lemma 3.3, we are now equipped to analyze the convergence behavior of SARA-based low-rank optimization. The theoretical result below demonstrates that SARA achieves provable convergence at a rate comparable to prior work, while the experimental results in the following section show improved empirical performance.

Theorem 3.4 (Convergence complexity of Low-rank MSGD with SARA, see Corollary A.6 for proof)**.** *By Assumption 3.1-3.2, if* $T \geq 2 + 128/(3\delta) + (128\sigma)^2/(9\sqrt{\delta}L\Delta)$ *for* $\Delta = f(x^{(0)}) - \inf_x f(x)$*, we choose* $\beta_1 = \left(1 + \sqrt{\frac{\delta^{3/2}\sigma^2 T}{L\Delta}}\right)^{-1}$, $\tau = \left\lceil \frac{64}{3\delta\beta_1}\right\rceil$*, and* $\eta = \left(4L + \sqrt{\frac{80L^2}{3\delta\beta_1^2} + \frac{80\tau^2L^2}{3\delta}} + \sqrt{\frac{16\tau L^2}{3\beta_1}}\right)^{-1}$*, low-rank MSGD-SARA with momentum re-projection converges as*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\nabla f(x^{(t)})\|_2^2\right] = \mathcal{O}\left(\frac{L\Delta}{\delta^{2.5}T} + \sqrt{\frac{L\Delta\sigma^2}{\delta^{3.5}T}}\right).$$

Comparing with [12], we adopt the same hyperparameters used in their study in Theorem 3.4. When examining the convergence rate, we note that the primary distinction lies in our convergence rate depends on δ (Theorem 3.4), whereas GoLore depends on $\underline{\delta} = \frac{r}{m}$ (Theorem 3.5).

Theorem 3.5 (Convergence of MSGD with GoLore, Corollary 3 of [12])**.** *Under Assumption 3.1-3.2, let every notation be defined as in Theorem 3.4, and using the same hyperparameters* β_1*,* τ*,* η*, Let* $\underline{\delta} = \frac{r}{m}$*. Then, MSGD-GoLore with momentum re-projection converges as*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\left\|\nabla f(x^{(t)})\right\|_F^2\right] = \mathcal{O}\left(\frac{L\Delta}{\underline{\delta}^{2.5}T} + \sqrt{\frac{L\Delta\sigma^2}{\underline{\delta}^{3.5}T}}\right),$$

Because SARA adopts importance sampling, we have $\delta < \underline{\delta}$. Thus, the convergence rate of MSGD-SARA is slower than MSGD-GoLore up to a constant factor. Compared to MSGD-GaLore (using dominant subspace), which does not have a provable convergence guarantee, SARA has the advantage in the theoretical convergence rate.

4 Experimental Results

In Section 4.1, we describe our experimental setup. In Section 4.2, we evaluate the efficacy of SARA when combined with various low-rank Adam optimizers. In Section 4.3, we show that SARA promotes subspace exploration and enables higher-rank updates. In Section 4.4, we further evaluate SARA on additional baselines and datasets.

4.1 Experiment Setting

In this section, we describe our experimental setup. We present the C4 dataset [34], architecture, and hyperparameters.

Pre-training on C4 Dataset. C4 [34], short for Colossal Clean Crawled Corpus, is a large-scale, open-source text dataset widely used in practice for pretraining transformer models such as BERT [32], T5 [39], and GPT models. C4 is also commonly used in the memory-efficient optimization community to evaluate the performance of memory-efficient optimizers [11, 42, 41, 12]. In our experiments, we pretrain LLaMA models of different sizes on the C4 dataset without data repetition, using a sufficiently large amount of data [13].

Architecture and Hyperparameters We evaluate the performance of different optimizers on LLaMA models with 60 million, 130 million, 350 million, and 1.1 billion parameters, using the same architecture as in [42]. For full-rank Adam, we use $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a learning rate of 0.001, except for the LLaMA-60M model, where the learning rate is set to 0.0025. More detailed hyperparameters for our re-implementation are provided in Appendix B. All experiments are conducted using one GPU node with 8 Nvidia A40 GPUs.

4.2 Efficacy of SARA with Low-Rank Adam Optimizers

In this section, we evaluate the efficacy of SARA with different low-rank Adam optimizers across multiple model sizes.

Table 1: Validation perplexity (PPL) of LLaMA models pretrained on the C4 dataset with 60M, 130M, and 350M parameters, comparing various low-rank optimizers with and without SARA. SARA consistently reduces the PPL gap relative to full-rank Adam, demonstrating its effectiveness across different optimizer variants and model scales.

	60M	**130M**	**350M**
Full-Rank Adam	27.71	23.27	18.21
GaLore-SARA-Adam	**30.47**	**24.21**	**19.16**
GaLore-Adam	31.50	24.88	19.68
PPL gap reduction	27.17%	41.61%	35.37%
Fira-SARA-Adam	**28.12**	**22.22**	**17.25**
Fira-Adam	28.42	22.37	17.35
PPL gap reduction	42.25%	—	—
GaLore-SARA-Adafactor	**30.06**	**24.09**	**18.88**
GaLore-Adafactor	31.13	24.79	19.45
PPL gap reduction	31.28%	46.05%	45.96%
GaLore-SARA-Adam-mini	**31.66**	**24.87**	**19.41**
GaLore-Adam-mini	32.08	25.46	19.89
PPL gap reduction	9.61%	26.94%	28.57%
GaLore-SARA-Adam (8bit)	**30.55**	**24.67**	**18.16**
GaLore-Adam (8bit)	31.62	25.35	18.63
PPL gap reduction	27.36%	32.69%	—
r/d_{model}	128/256	256/768	256/1024
Tokens	1.5B	2.2B	6B

Efficacy of SARA with different low-rank Adam optimizers First, we evaluate the efficacy of SARA when combined with various low-rank Adam optimizers. Table 1 shows that SARA consistently outperforms dominant subspace selection. In cases where full-rank Adam achieves the lowest PPL, we also report the percentage reduction in the PPL gap achieved by SARA compared to the dominant subspace baseline. As shown in Table 1, SARA reduces the PPL gap by up to 46.05%. In scenarios where full-rank Adam does not achieve the lowest PPL, SARA still improves performance over leading singular vector selection. SARA proves effective not only with low-rank Adam variants such as GaLore-Adam and Fira-Adam, but also with optimizers that approximate second moments, e.g., GaLore-Adafactor and GaLore-Adam-mini. Results with the 8-bit optimizer further highlight the robustness of SARA under low-precision optimizer state storage.

Scale Up to Llama-1.1B We also evaluate the efficacy of SARA on the pretraining of LLaMA-1.1B. Due to limited computational resources, we conduct experiments using only GaLore-Adam. As shown in Table 2, SARA remains effective on LLaMA-1.1B.

Table 2: PPL on LLaMA-1.1B pretrained with full-rank Adam, GaLore-Adam, and GaLore-SARA-Adam on the C4 dataset. Despite the larger model size, SARA continues to outperform dominant subspace selection, confirming its scalability and robustness.

	Full	GaLore-SARA-Adam	GaLore-Adam
1.1B	15.90	**15.36**	15.47
r/d_{model}	512/2048	512/2048	512/2048
Tokens	13.4B	13.4B	13.4B

4.3 SARA Encourages Subspace Exploration and Higher-Rank Updates

In this section, we empirically show that SARA encourages subspace exploration and enables higher-rank updates.

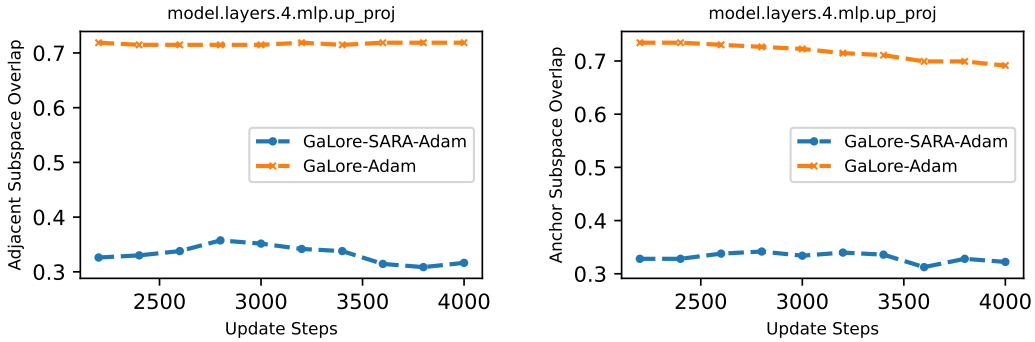


Figure 3: a). The left figure shows the overlap between adjacent subspaces in GaLore-Adam and GaLore-SARA-Adam during pretraining on the LLaMA-60M model between 2200-th and 4000-th iteration. b). The right figure takes the low-rank subspace at the 2000-th iteration as the anchor subspace, and shows the overlap between subspaces in later iterations and the anchor subspace.

SARA encourages subspace exploration [41] provides an interesting observation that the similarity between adjacent subspaces in some layers gradually becomes very high during pretraining, we observe a similar phenomenon shown in Figure 2. We adopt a metric to measure overlap between two subspaces from [9]. Given two orthonormal matrices $U, V \in \mathbb{R}^{m\times r}$, we have

$$U^T U = V^T V = I_r,$$

the overlap between two subspaces spanned by U and V are defined as

$$\text{overlap}(U, V) = \frac{1}{r}\sum_{i=1}^{r} \|U^T V_{:,i}\|_2^2,$$

where $V_{:,i}$ denotes the i-th column of V. We adopt the above metric to show that the observation in [41] is not because of using cosine similarity as the measure, but the frozen subspace phenomenon also exists when using other metrics to measure subspace overlap (or subspace similarity).

An interesting fact is that the overlap between adjacent subspaces in GaLore-SARA-Adam is much lower than GaLore-Adam, as shown in Figure 3 (a). In Figure 3 (b), we chose the subspace at the 2000-th iteration as the anchor subspace and examined the overlap between subspaces from later iterations, specifically between the 2200-th and 4000-th iterations. We observe that the overlap between the anchor subspace and later subspaces of GaLore-SARA-Adam is lower than that of GaLore-Adam. This indicates that SARA encourages the optimization trajectory to explore more different subspaces compared to using the dominant subspace.

SARA Enables Higher-rank Update Figure 4 shows that the update produced by SARA in the weight matrix exhibits more evenly distributed singular values compared to the update using the dominant subspace. This suggests that SARA helps overcome the low-rank bottleneck associated with the dominant subspace approach. We observe both this higher-rank update and improved subspace exploration occurring simultaneously. We believe these two phenomena are correlated. One possible explanation is that better exploration of diverse subspaces leads to a higher-rank update.

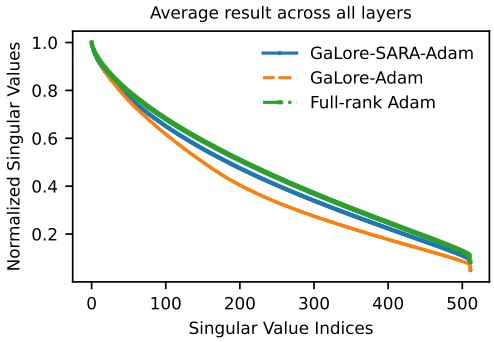


Figure 4: The average result of normalized singular values of the weight difference between the 28k-step checkpoint and 30k-step checkpoint across all layers in LLaMA-60M model during pretraining.

4.4 Additional Baselines and Datasets

In this section, we present additional baselines (GoLore [12] and online PCA [24]) and a dataset (SlimPajama).

More Baselines for Pretraining on C4 To provide a more comprehensive evaluation of SARA, we conduct extensive pretraining experiments on the C4 dataset with additional baseline methods. Beyond the comparisons shown in our main results, we include two particularly relevant baselines: GoLore [12] and the online PCA approach from [24]. These baselines were selected because they are also competitive alternatives to our method. The results, presented in Table 3, reveal several important insights. First, we observe that GoLore-Adam performs reasonably well, achieving validation perplexity of 31.61 and 24.01 for the 60M and 130M parameter models respectively. However, SARA consistently outperforms GoLore by significant margins (1.14 and 0.93 perplexity points for the two model sizes), demonstrating the effectiveness of our approach in learning more efficient low-rank representations. The comparison with [24]'s online PCA method is particularly illuminating. Their approach, while computationally efficient, shows substantially higher perplexity (33.69 and 30.62) compared to both GoLore and SARA. This is mainly due to the unstable training loss curve during the online PCA method pretraining.

Table 3: Validation perplexity comparison on the C4 dataset for LLaMA models with 60M and 130M parameters using additional baselines.

C4 Dataset (validation perplexity)	60M (1.5B tokens)	130M (3B tokens)
GoLore-Adam	31.61	24.01
[24] with Adam	33.69	30.62
GaLore-SARA-Adam	30.47	23.08
Full rank Adam	27.71	22.19

Pretraining Results on More Datasets To demonstrate the generalizability of SARA beyond the C4 dataset, we conduct additional pretraining experiments on the SlimPajama dataset, a carefully filtered and deduplicated subset of the Pile corpus. The results in Table 4 confirm that our method's advantages are not dataset-specific. Several interesting patterns emerge from the SlimPajama results. First, the performance gaps between methods are slightly smaller on SlimPajama compared to C4. For the 130M parameter model, SARA improves upon standard GaLore-Adam by 0.57 perplexity points (25.23 vs 25.80), while trailing full-rank Adam by only 0.23 points. This suggests that our method may be particularly effective on higher-quality, deduplicated datasets like SlimPajama. We also note that the absolute perplexity values are slightly lower on SlimPajama than on C4 for comparable model sizes, which may reflect the dataset's more careful curation. This makes the strong performance of SARA even more noteworthy, as it demonstrates effectiveness across different difficulty levels.

Additional experiments on high-rank updates, anchor similarity, and adjacent overlap can be found in Appendix E.1, E.2, and E.3, respectively.

Table 4: Validation perplexity on the SlimPajama dataset for LLaMA models with 60M and 130M parameters.

SlimPajama (validation perplexity)	60M	130M
Full rank Adam	27.79	25.00
GaLore-Adam	31.76	25.80
GaLore-SARA-Adam	30.79	25.23

5 Related Work

Memory Efficient Parametrization. LoRA [14] can be seen as a memory-efficient parametrization of weights in LLMs and is widely used in fine-tuning. LoRA's bottleneck lies in its low-rank structure, which impedes its expressiveness. COLA [38], Delta-LoRA [44], and PLoRA [27] propose to increase the rank and improve the performance of LoRA. ReLoRA [23] and SLTrain [10] extend LoRA to pre-training tasks by merging and resetting adapters, and adopting low-rank plus sparse parameterization, respectively. MoRA [18] alleviates the shortcoming of the low-rank disadvantage of LoRA by sharing the same trainable parameters to achieve a higher-rank update. Additionally, [25] analyzes the sparsity-based parameter-efficient fine-tuning (SPEFT) for LLMs, which is an alternative method of LoRA. [31] designs a novel method for memory-efficient fine-tuning for LLMs. It has been shown that it can outperform LoRA and full-parameter training in many cases. Similarly, [15] proposes another novel fine-tuning method called Half Fine-Tuning (HFT), which can mitigate "catastrophic forgetting" in LLMs during sequential training and instruction tuning. Finally, [22] also proposes a novel memory-efficient fine-tuning method by strategically selecting layers to update based on outlier statistics. Our paper also considers the memory and convergence behavior, but the difference is that we mainly focus on the LLM pretraining instead of fine-tuning.

Memory Efficient Optimizer. One way to achieve memory-efficient optimization is by using memory-efficient optimizers, which primarily aim to reduce the memory cost of optimizer states in Adam [19]. A series of works [35, 40, 26, 43] factorize the second moment in Adam. Quantizing optimizer states and storing them in low-precision formats has also proven successful [21, 7]. Another line of work focuses on gradient compression methods. GaLore [42] and Q-GaLore [41] use SVD to apply dense low-rank projections to gradients. FLora [11] and GoLore [12] adopt random projection, while Grass [28] employs sparse low-rank projection to gradients.

Subspace Learning. Existing studies provide sophisticated analyses of various subspace learning algorithms [6, 20, 16]. [9] claims that gradient descent primarily occurs in the dominant subspace, which is spanned by the top eigenvectors of the Hessian. In contrast, [36] argues that, due to noise in SGD, the alignment between the gradient and the dominant subspace is spurious, and learning does not occur in the dominant subspace but rather in its orthogonal complement, i.e., the bulk subspace. Intuitively, our findings align with those of [36], suggesting that selecting basis vectors based on specific sampling probabilities can enhance the performance of LLMs during pre-training.

6 Conclusion

In this paper, we propose SARA for low-rank optimization in LLM pretraining. The motivation is to find an effective subspace selection method to overcome the low-rank bottleneck caused by the frozen dominant subspace in low-rank optimization. SARA samples singular vectors of mini-batch gradients with probabilities proportional to their singular values, this enables optimization trajectory to explore more different subspaces. Theoretically, in Theorem 3.4, we show that GaLore-SARA-MSGD achieves comparison convergence rate as GoLore-MSGD, which is $\mathcal{O}\left(\frac{L\Delta}{\delta^{2.5}T}+\sqrt{\frac{L\Delta\sigma^2}{\delta^{3.5}T}}\right)$. Empirically, we find that SARA improves the language modeling capability of pretrained models compared to using the dominant subspace, as verified by experiments involving SARA and dominant subspace selection with multiple low-rank optimizers.

Acknowledgment

Lin F. Yang is supported in part by NSF Grant 2221871 and an Amazon Faculty Award.

References

[1] Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[3] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[4] Xi Chen, Kaituo Feng, Changsheng Li, Xunhao Lai, Xiangyu Yue, Ye Yuan, and Guoren Wang. Fira: Can we achieve full-rank training of llms under low-rank constraint? *arXiv preprint arXiv:2410.01623*, 2024.

[5] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

[6] Romain Cosson, Ali Jadbabaie, Anuran Makur, Amirhossein Reisizadeh, and Devavrat Shah. Low-rank gradient descent. *IEEE Open Journal of Control Systems*, 2023.

[7] Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 8-bit optimizers via block-wise quantization. *arXiv preprint arXiv:2110.02861*, 2021.

[8] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[9] Guy Gur-Ari, Daniel A Roberts, and Ethan Dyer. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*, 2018.

[10] Andi Han, Jiaxiang Li, Wei Huang, Mingyi Hong, Akiko Takeda, Pratik Jawanpuria, and Bamdev Mishra. Sltrain: a sparse plus low-rank approach for parameter and memory efficient pretraining. *arXiv preprint arXiv:2406.02214*, 2024.

[11] Yongchang Hao, Yanshuai Cao, and Lili Mou. Flora: Low-rank adapters are secretly gradient compressors. *arXiv preprint arXiv:2402.03293*, 2024.

[12] Yutong He, Pengrui Li, Yipeng Hu, Chuyan Chen, and Kun Yuan. Subspace optimization for large language models with convergence guarantees. *arXiv preprint arXiv:2410.11289*, 2024.

[13] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. An empirical analysis of compute-optimal large language model training. *Advances in Neural Information Processing Systems*, 35:30016–30030, 2022.

[14] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[15] Tingfeng Hui, Zhenyu Zhang, Shuohuan Wang, Weiran Xu, Yu Sun, and Hua Wu. HFT: Half fine-tuning for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025.

[16] Ali Jadbabaie, Anuran Makur, and Amirhossein Reisizadeh. Adaptive low-rank gradient descent. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 3315–3320. IEEE, 2023.

[17] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[18] Ting Jiang, Shaohan Huang, Shengyue Luo, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, et al. Mora: High-rank updating for parameter-efficient fine-tuning. *arXiv preprint arXiv:2405.12130*, 2024.

[19] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[20] David Kozak, Stephen Becker, Alireza Doostan, and Luis Tenorio. Stochastic subspace descent. *arXiv preprint arXiv:1904.01145*, 2019.

[21] Bingrui Li, Jianfei Chen, and Jun Zhu. Memory efficient optimizers with 4-bit states. *Advances in Neural Information Processing Systems*, 36, 2024.

[22] Pengxiang Li, Lu Yin, Xiaowei Gao, and Shiwei Liu. Outlier-weighed layerwise sampling for llm fine-tuning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 19460–19473, 2025.

[23] Vladislav Lialin, Sherin Muckatira, Namrata Shivagunde, and Anna Rumshisky. Relora: High-rank training through low-rank updates. In *The Twelfth International Conference on Learning Representations*, 2023.

[24] Kaizhao Liang, Bo Liu, Lizhang Chen, and qiang liu. Memory-efficient LLM training with online subspace descent. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[25] Xinxin Liu, Aaron Thomas, Cheng Zhang, Jianyi Cheng, Yiren Zhao, and Xitong Gao. Refining salience-aware sparse fine-tuning strategies for language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025.

[26] Yang Luo, Xiaozhe Ren, Zangwei Zheng, Zhuo Jiang, Xin Jiang, and Yang You. Came: Confidence-guided adaptive memory efficient optimization. *arXiv preprint arXiv:2307.02047*, 2023.

[27] Xiangdi Meng, Damai Dai, Weiyao Luo, Zhe Yang, Shaoxiang Wu, Xiaochen Wang, Peiyi Wang, Qingxiu Dong, Liang Chen, and Zhifang Sui. Periodiclora: Breaking the low-rank bottleneck in lora optimization. *arXiv preprint arXiv:2402.16141*, 2024.

[28] Aashiq Muhamed, Oscar Li, David Woodruff, Mona Diab, and Virginia Smith. Grass: Compute efficient low-memory llm training with structured sparse gradients. *arXiv preprint arXiv:2406.17660*, 2024.

[29] Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, et al. Efficient large-scale language model training on gpu clusters using megatron-lm. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15, 2021.

[30] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[31] Rui Pan, Xiang Liu, Shizhe Diao, Renjie Pi, Jipeng Zhang, Chi Han, and Tong Zhang. Lisa: Layerwise importance sampling for memory-efficient large language model fine-tuning. *Advances in Neural Information Processing Systems*, 37:57018–57049, 2024.

[32] Jacob Portes, Alexander Trott, Sam Havens, Daniel King, Abhinav Venigalla, Moin Nadeem, Nikhil Sardana, Daya Khudia, and Jonathan Frankle. Mosaicbert: A bidirectional encoder optimized for fast pretraining. *Advances in Neural Information Processing Systems*, 36:3106–3130, 2023.

[33] Alec Radford. Improving language understanding by generative pre-training. 2018.

[34] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[35] Noam Shazeer and Mitchell Stern. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR, 2018.

[36] Minhak Song, Kwangjun Ahn, and Chulhee Yun. Does sgd really happen in tiny subspaces? *arXiv preprint arXiv:2405.16002*, 2024.

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

[38] Wenhan Xia, Chengwei Qin, and Elad Hazan. Chain of lora: Efficient fine-tuning of language models via residual learning. *arXiv preprint arXiv:2401.04151*, 2024.

[39] L Xue. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*, 2020.

[40] Yushun Zhang, Congliang Chen, Ziniu Li, Tian Ding, Chenwei Wu, Yinyu Ye, Zhi-Quan Luo, and Ruoyu Sun. Adam-mini: Use fewer learning rates to gain more. *arXiv preprint arXiv:2406.16793*, 2024.

[41] Zhenyu Zhang, Ajay Jaiswal, Lu Yin, Shiwei Liu, Jiawei Zhao, Yuandong Tian, and Zhangyang Wang. Q-galore: Quantized galore with int4 projection and layer-adaptive low-rank gradients. *arXiv preprint arXiv:2407.08296*, 2024.

[42] Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. Galore: Memory-efficient llm training by gradient low-rank projection. *arXiv preprint arXiv:2403.03507*, 2024.

[43] Pengxiang Zhao, Ping Li, Yingjie Gu, Yi Zheng, Stephan Ludger Kölker, Zhefeng Wang, and Xiaoming Yuan. Adapprox: Adaptive approximation in adam optimization via randomized low-rank matrices. *arXiv preprint arXiv:2403.14958*, 2024.

[44] Bojia Zi, Xianbiao Qi, Lingzhi Wang, Jianan Wang, Kam-Fai Wong, and Lei Zhang. Delta-lora: Fine-tuning high-rank parameters with the delta of low-rank matrices. *arXiv preprint arXiv:2309.02411*, 2023.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: We have explicitly presented our main claims in our abstract and introduction (see Section 1).

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We have detailedly discussed the limitations of our paper in Section C.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

Justification: The assumptions we have can be found at Assumption 3.1 and Assumption 3.2. The proof of Theorem 3.4 can be found in Appendix A. The proof of Lemma 3.3 can be found in Appendix A.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: We have provided the pseudocode for our algorithms—SARA (Algorithm 2) — as well as details on the LLaMA models, datasets, and evaluation metrics. The hyperparameters is included in Appendix B. The GitHub Repo will be open to public after the review session.

Guidelines:

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

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

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

   Answer: [No]

   Justification: The datasets we used are public and can be accessed by everyone. Also, we will release our code after the review session.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We provide the training and test details, hyperparameters, experiment settings, etc (see Section 4.1 and Appendix B). Once the review session is done, we will provide the code of our experiments.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: We do not have error bars.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide this information in Section 4.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: All authors have reviewed the NeurIPS Code of Ethics and confirm that the research presented in this paper fully complies with it.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We have discussed the positive and negative societal impact in Section D.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [NA]

    Justification: Our paper poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We cited the previous papers that produced the code package or dataset.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [NA]

    Justification: We do not release new assets.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: We do not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: We not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [Yes]

    Justification: We only use LLM to check our grammar.

    Guidelines:

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

Appendix

A Proof Details for SARA

Lemma A.1 (Gradient connections, Lemma 2 in [12]). *Let $\nabla_\ell f(x)$ denote the gradient with respect to the ℓ-th layer parameters at point x, and suppose $x^{(t)}$ denotes the parameters at iteration t. Then for any integers $t \geq 0$ and $\tau > 0$, we have*

Then, for all $t, \tau > 0$, we have

$$\|\nabla_\ell f(\mathbf{x}^{(t)})\|_F^2 \leq \frac{2}{\tau}\sum_{r=0}^{\tau-1}\|\nabla_\ell f(\mathbf{x}^{(t+r)})\|_F^2 + (\tau-1)\sum_{r=0}^{\tau-2}\left\|\nabla_\ell f(\mathbf{x}^{(t+r+1)}) - \nabla_\ell f(\mathbf{x}^{(t+r)})\right\|_F^2.$$

Lemma A.2 (Error of SARA's Projection). *Let τ be the update period of SARA, and r be the rank of low-rank subspace in SARA. For all $i \in [m], l \in [N], k \in \mathbb{N}$, let $p_l^{(t)}(i)$ denote the probability that the i-th basis vector is selected for the l-th layer at time $t = k\tau$, and define*

$$\delta_l^{(t)} := \min_{i\in[m]} p_l^{(t)}(i), \quad \delta := \min_{l\in[m],t\geq 0}\delta_l^{(t)}$$

Let $P_l^{(t)} \in \mathbb{R}^{m\times r}$ denote the orthonormal projection matrix and let $\nabla_l f(x^{(t)}) \in \mathbb{R}^{m\times n_l}$ be the gradient matrix of the l-th layer at time t. Then, the following inequality holds:

$$\mathbb{E}\left[\left\|\left(I - P_l^{(t)}(P_l^{(t)})^\top\right)\nabla_l f(x^{(t)})\right\|_F^2\right] \leq (1-\delta)\cdot\mathbb{E}\left[\left\|\nabla_l f(x^{(t)})\right\|_F^2\right].$$

Proof. We analyze the expected projection residual:

$$\begin{aligned}
&\mathbb{E}\left[\left\|\left(I - P_l^{(t)}(P_l^{(t)})^\top\right)\nabla_l f(x^{(t)})\right\|_F^2\right] \\
&= \mathbb{E}_{\nabla_l f(x^{(t)})}\left[\mathbb{E}_{P_l^{(t)}}\left[\left\|\left(I - P_l^{(t)}(P_l^{(t)})^\top\right)\nabla_l f(x^{(t)})\right\|_F^2 \,\middle|\, \nabla_l f(x^{(t)})\right]\right] \\
&= \mathbb{E}_{\nabla_l f(x^{(t)})}\left[\operatorname{tr}\left(\mathbb{E}_{P_l^{(t)}}\left[\left(I - P_l^{(t)}(P_l^{(t)})^\top\right)^2\right]\cdot\nabla_l f(x^{(t)})\nabla_l f(x^{(t)})^\top\right)\right].
\end{aligned}$$

Let $\{U_j\}_{j=1}^m$ be a fixed orthonormal basis for $\mathbb{R}^m$, and define the indicator variable $\mathbf{1}_{\{j\}}$ to denote whether U_j is selected. Then,

$$\mathbb{E}_{P_l^{(t)}}\left[I - P_l^{(t)}(P_l^{(t)})^\top\right] = \sum_{j=1}^m (1 - \mathbb{E}[\mathbf{1}_{\{j\}}])U_j U_j^\top.$$

Therefore,

$$\begin{aligned}
\mathbb{E}\left[\left\|\left(I - P_l^{(t)}(P_l^{(t)})^\top\right)\nabla_l f(x^{(t)})\right\|_F^2\right] &= \mathbb{E}\left[\operatorname{tr}\left(\sum_{j=1}^m (1-\mathbb{E}[\mathbf{1}_{\{j\}}])U_j U_j^\top\cdot\nabla_l f(x^{(t)})\nabla_l f(x^{(t)})^\top\right)\right] \\
&= \sum_{j=1}^m (1-\mathbb{E}[\mathbf{1}_{\{j\}}])\cdot\mathbb{E}\left[\left\|U_j^\top\nabla_l f(x^{(t)})\right\|_2^2\right] \\
&\leq (1-\min_j \mathbb{E}[\mathbf{1}_{\{j\}}])\cdot\mathbb{E}\left[\left\|\nabla_l f(x^{(t)})\right\|_F^2\right] \\
&= (1-\delta_l^{(t)})\cdot\mathbb{E}\left[\left\|\nabla_l f(x^{(t)})\right\|_F^2\right] \\
&\leq (1-\delta)\cdot\mathbb{E}\left[\left\|\nabla_l f(x^{(t)})\right\|_F^2\right].
\end{aligned}$$

□

Lemma A.3 (Momentum Contraction of SARA). *For SARA with momentum re-projection, for all $l \in [N]$, we have the inequalities below hold.*

- ***Part 1*** **($t = 0$).**

$$\mathbb{E}\left[\left\|\widetilde{M}_l^{(0)} - \nabla_l f(x^{(0)})\right\|_F^2\right] \leq \left(1 - (2\beta_1 - \beta_1^2)\delta\right)\mathbb{E}\left[\left\|\nabla_l f(x^{(0)})\right\|_F^2\right] + \beta_1^2\sigma_l^2.$$

- ***Part 2*** **($t = k\tau$, $k \in \mathbb{N}$).**

$$\begin{aligned}
&\mathbb{E}\left[\left\|\widetilde{M}_l^{(t)} - \nabla_l f(x^{(t)})\right\|_F^2\right] - \left(1 - \left(1 - \frac{\delta}{4}\right)\beta_1\right)\mathbb{E}\left[\left\|\widetilde{M}_l^{(t-1)} - \nabla_l f(x^{(t-1)})\right\|_F^2\right] \\
&\leq \frac{2(1-\delta)}{\tau}\sum_{r=0}^{\tau-1}\mathbb{E}\left[\left\|\nabla_l f(x^{(k\tau+r)})\right\|_F^2\right] + \frac{5(1-\beta_1)}{\beta_1\delta}\mathbb{E}\left[\left\|\nabla_l f(x^{(t)}) - \nabla_l f(x^{(t-1)})\right\|_F^2\right] \\
&\quad + (\tau-1)(1-\delta)\sum_{r=0}^{\tau-2}\mathbb{E}\left[\left\|\nabla_l f(x^{(k\tau+r+1)}) - \nabla_l f(x^{(k\tau+r)})\right\|_F^2\right] + \beta_1^2\sigma_l^2.
\end{aligned}$$

- ***Part 3*** **($t = k\tau + r$, $1 \leq r \leq \tau - 1$).**

$$\begin{aligned}
&\mathbb{E}\left[\left\|\widetilde{M}_l^{(t)} - \nabla_l f(x^{(t)})\right\|_F^2\right] - \left(1 - \left(1 - \frac{\delta}{4}\right)\beta_1\right)\mathbb{E}\left[\left\|\widetilde{M}_l^{(t-1)} - \nabla_l f(x^{(t-1)})\right\|_F^2\right] \\
&\leq \left(1 - \frac{\delta}{2}\right)\beta_1\mathbb{E}\left[\left\|\nabla_l f(x^{(t)})\right\|_F^2\right] + \frac{5(1-\beta_1)}{\beta_1\delta}\mathbb{E}\left[\left\|\nabla_l f(x^{(t)}) - \nabla_l f(x^{(t-1)})\right\|_F^2\right] \\
&\quad + \frac{10r\beta_1}{\delta}\sum_{i=1}^{r}\mathbb{E}\left[\left\|\nabla_l f(x^{(k\tau+i)}) - \nabla_l f(x^{(k\tau+i-1)})\right\|_F^2\right] + \beta_1^2\sigma_l^2.
\end{aligned}$$

Proof. **Proof of Part 1.**

Suppose $t = 0$. By definition of the momentum estimator $\widetilde{M}_l^{(0)} = \beta_1 P_l^{(0)}(P_l^{(0)})^\top G_l^{(0)}$, we decompose the error as:

$$\begin{aligned}
\mathbb{E}\left[\left\|\widetilde{M}_l^{(0)} - \nabla_l f(x^{(0)})\right\|_F^2\right] &= \mathbb{E}\left[\left\|\beta_1 P_l^{(0)}(P_l^{(0)})^\top\left(G_l^{(0)} - \nabla_l f(x^{(0)})\right)\right\|_F^2\right] \\
&\quad + \mathbb{E}\left[\left\|\left(\beta_1 P_l^{(0)}(P_l^{(0)})^\top - I\right)\nabla_l f(x^{(0)})\right\|_F^2\right].
\end{aligned}$$

The first term is bounded using the assumption that $G_l^{(0)}$ is an unbiased estimator with variance σ_l^2:

$$\mathbb{E}\left[\left\|\beta_1 P_l^{(0)}(P_l^{(0)})^\top\left(G_l^{(0)} - \nabla_l f(x^{(0)})\right)\right\|_F^2\right] \leq \beta_1^2\sigma_l^2.$$

For the second term, we apply Lemma A.2:

$$\mathbb{E}\left[\left\|\left(I - \beta_1 P_l^{(0)}(P_l^{(0)})^\top\right)\nabla_l f(x^{(0)})\right\|_F^2\right] \leq \left(1 - (2\beta_1 - \beta_1^2)\delta\right)\mathbb{E}\left[\left\|\nabla_l f(x^{(0)})\right\|_F^2\right].$$

Combining both bounds:

$$\mathbb{E}\left[\left\|\widetilde{M}_l^{(0)} - \nabla_l f(x^{(0)})\right\|_F^2\right] \leq \left(1 - (2\beta_1 - \beta_1^2)\delta\right)\mathbb{E}\left[\left\|\nabla_l f(x^{(0)})\right\|_F^2\right] + \beta_1^2\sigma_l^2.$$

Proof of Part 2.

Suppose $t = k\tau$. At the projection step, we have the update rule:

$$\widetilde{M}_l^{(t)} = P_l^{(t)}(P_l^{(t)})^\top\left((1-\beta_1)\widetilde{M}_l^{(t-1)} + \beta_1 G_l^{(t)}\right).$$

Then:

$$\mathbb{E}\left[\left\|\widetilde{M}_l^{(t)} - \nabla_l f(x^{(t)})\right\|_F^2\right] = \mathbb{E}\left[\left\|P_l^{(t)}(P_l^{(t)})^\top\left((1-\beta_1)\widetilde{M}_l^{(t-1)} + \beta_1 G_l^{(t)} - \nabla_l f(x^{(t)})\right)\right. \quad (1)$$

$$- \left(I - P_l^{(t)}(P_l^{(t)})^\top\right) \nabla_l f(x^{(t)})\Big\|_F^2\Big] \tag{2}$$

$$= \mathbb{E}\left[\left\|P_l^{(t)}(P_l^{(t)})^\top \left((1-\beta_1)\widetilde{M}_l^{(t-1)} + \beta_1 G_l^{(t)} - \nabla_l f(x^{(t)})\right)\right\|_F^2\right]$$

$$+ \mathbb{E}\left[\left\|\left(I - P_l^{(t)}(P_l^{(t)})^\top\right) \nabla_l f(x^{(t)})\right\|_F^2\right]. \tag{3}$$

The second equality is because of the Pythagorean Theorem.

Applying Lemma A.2 to the second term:

$$\mathbb{E}\left[\left\|\left(I - P_l^{(t)}(P_l^{(t)})^\top\right) \nabla_l f(x^{(t)})\right\|_F^2\right] \leq (1-\delta)\mathbb{E}\left[\left\|\nabla_l f(x^{(t)})\right\|_F^2\right].$$

The first term is bounded as below:

$$\begin{aligned}
&\mathbb{E}\left[\left\|P_l^{(t)}(P_l^{(t)})^\top \left((1-\beta_1)\widetilde{M}_l^{(t-1)} + \beta_1 G_l^{(t)} - \nabla_l f(x^{(t)})\right)\right\|_F^2\right] \\
&\leq \mathbb{E}\left[\left\|(1-\beta_1)\widetilde{M}_l^{(t-1)} + \beta_1 G_l^{(t)} - \nabla_l f(x^{(t)})\right\|_F^2\right] \\
&= \mathbb{E}\left[\left\|(1-\beta_1)(\widetilde{M}_l^{(t-1)} - \nabla_l f(x^{(t)})) + \beta_1(G_l^{(t)} - \nabla_l f(x^{(t)}))\right\|_F^2\right] \\
&\leq (1-\beta_1)\mathbb{E}\left[\left\|\widetilde{M}_l^{(t-1)} - \nabla_l f(x^{(t)})\right\|_F^2\right] + \beta_1 \mathbb{E}\left[\left\|G_l^{(t)} - \nabla_l f(x^{(t)})\right\|_F^2\right] \\
&\leq (1-\beta_1)\mathbb{E}\left[\left\|\widetilde{M}_l^{(t-1)} - \nabla_l f(x^{(t)})\right\|_F^2\right] + \beta_1 \sigma_l^2,
\end{aligned} \tag{4}$$

where the first inequality is because of $\left\|P_l^{(t)}(P_l^{(t)})^T\right\|_2 = 1$, the second inequality is because of the independent noise of mini-batch gradient, the third inequality is because of the bounded noise assumption.

To bound $\mathbb{E}\left[\left\|\widetilde{M}_l^{(t-1)} - \nabla_l f(x^{(t)})\right\|_F^2\right]$, we apply Young's inequality:

$$\begin{aligned}
&\mathbb{E}\left[\left\|\widetilde{M}_l^{(t-1)} - \nabla_l f(x^{(t)})\right\|_F^2\right] \\
&\leq \left(1 + \frac{\delta\beta_1}{4}\right)\mathbb{E}\left[\left\|\widetilde{M}_l^{(t-1)} - \nabla_l f(x^{(t-1)})\right\|_F^2\right] + \left(1 + \frac{4}{\delta\beta_1}\right)\mathbb{E}\left[\left\|\nabla_l f(x^{(t)}) - \nabla_l f(x^{(t-1)})\right\|_F^2\right]
\end{aligned} \tag{5}$$

So far, we can have the upper bound for the first term in Eq. (4). By applying Eq. (5), Eq. (4), and Lemma A.1 which naturally holds in our problem setting, we have shown **Part 2**.

Proof of Part 3.

Suppose $t = k\tau + r$, $1 \leq r \leq \tau - 1$). In this case, $P_l^{(t)} = P_l^{(k\tau)}$ is reused. The update becomes:

$$\widetilde{M}_l^{(t)} = (1-\beta_1)\widetilde{M}_l^{(t-1)} + \beta_1 P_l^{(t)}(P_l^{(t)})^\top G_l^{(t)}.$$

Using the standard decomposition:

$$\begin{aligned}
\widetilde{M}_l^{(t)} - \nabla_l f(x^{(t)}) = (1-\beta_1)(\widetilde{M}_l^{(t-1)} - \nabla_l f(x^{(t)})) + \beta_1(P_l^{(t)}(P_l^{(t)})^\top - I)\nabla_l f(x^{(t)}) \\
+ \beta_1 P_l^{(t)}(P_l^{(t)})^\top (G_l^{(t)} - \nabla_l f(x^{(t)})).
\end{aligned}$$

By unbiasedness, we have

$$\mathbb{E}\left[\left\|\widetilde{M}_l^{(t)} - \nabla_l f(x^{(t)})\right\|_F^2\right]$$

$$= \mathbb{E}\left[\left\|(1-\beta_1)(\widetilde{M}_l^{(t-1)} - \nabla_l f(x^{(t)})) + \beta_1(P_l^{(t)}(P_l^{(t)})^\top - I)\nabla_l f(x^{(t)})\right\|_F^2\right]$$
$$+ \mathbb{E}\left[\left\|\beta_1 P_l^{(t)}(P_l^{(t)})^\top(G_l^{(t)} - \nabla_l f(x^{(t)}))\right\|_F^2\right].$$

Recall that by $\left\|P_l^{(t)}(P_l^{(t)})^T\right\|_2 = 1$ and the noise assumption, so we have

$$\mathbb{E}\left[\left\|\beta_1 P_l^{(t)}(P_l^{(t)})^\top(G_l^{(t)} - \nabla_l f(x^{(t)}))\right\|_F^2\right] \le \beta_1^2\sigma_l^2.$$

By applying Jensen's inequality, we have

$$\mathbb{E}\left[\left\|\widetilde{M}_l^{(t)} - \nabla_l f(x^{(t)})\right\|_F^2\right] \le (1-\beta_1)\mathbb{E}\left[\left\|\widetilde{M}_l^{(t-1)} - \nabla_l f(x^{(t)})\right\|_F^2\right]$$
$$+ \beta_1\mathbb{E}\left[\left\|(P_l^{(t)}(P_l^{(t)})^\top - I)\nabla_l f(x^{(t)})\right\|_F^2\right] + \beta_1^2\sigma_l^2.$$

We now bound the projection error using Lemma A.2:

$$\begin{aligned}\mathbb{E}\left[\left\|(P_l^{(t)}(P_l^{(t)})^\top - I)\nabla_l f(x^{(t)})\right\|_F^2\right] \le& \left(1+\frac{\delta}{4}\right)\mathbb{E}\left[\left\|(P_l^{(t)}(P_l^{(t)})^\top - I)\nabla_l f(x^{(k\tau)})\right\|_F^2\right]\\ &+ \left(1+\frac{4}{\delta}\right)\mathbb{E}\left[\left\|\nabla_l f(x^{(t)}) - \nabla_l f(x^{(k\tau)})\right\|_F^2\right]\\ \le& \left(1-\frac{3\delta}{4}\right)\mathbb{E}\left[\left\|\nabla_l f(x^{(k\tau)})\right\|_F^2\right]\\ &+ \left(1+\frac{4}{\delta}\right)\mathbb{E}\left[\left\|\nabla_l f(x^{(t)}) - \nabla_l f(x^{(k\tau)})\right\|_F^2\right].\end{aligned}$$

The first equality is because of Young's inequality, the second inequality is because of Lemma A.2.

This concludes the bound. The final contraction inequality is then followed by applying this bound and collecting terms. □

For **Part 3** of Lemma A.3, we get exactly the same bound for our SARA compared with their GoLore.

Though our Momentum Contraction result is a little worse than the one in [12], we can still get the same result for Momentum Error Bound, as shown in Lemma A.4.

Lemma A.4 (Momentum Error Bound of MSGD with SARA)**.** *Define*

$$\sigma^2 = \sum_{l\in[N]}\sigma_l^2$$

Then we have

$$\begin{aligned}&\sum_{t=0}^{K\tau-1}\mathbb{E}\left[\left\|\widetilde{M}^{(t)} - \nabla f(x^{(t)})\right\|_F^2\right]\\ \le& \left(\frac{5(1-\beta_1)}{(1-\delta/4)\delta\beta_1^2} + \frac{5\tau(1-\tau)}{(1-\delta/4)\delta} + \frac{\tau-1}{(1-\delta/4)\beta}\right)L^2\sum_{t=0}^{K\tau-2}\mathbb{E}\left[\left\|x^{(t+1)} - x^{(t)}\right\|_F^2\right]\\ &+ \left(\frac{1-\delta/2}{1-\delta/4} + \frac{2}{(1-\delta/4)\tau\beta_1}\right)\sum_{t=0}^{K\tau-2}\mathbb{E}\left[\left\|\nabla f(x^{(t)})\right\|_F^2\right] + \frac{K\tau\beta_1\sigma^2}{1-\delta/4}\end{aligned}$$

Proof. First we apply summation to **Part 3** of Lemma A.3 as follows:

$$\sum_{t=k\tau+1}^{(k+1)\tau-1}\mathbb{E}\left[\left\|\widetilde{M}_l^{(t)} - \nabla_l f(x^{(t)})\right\|_F^2\right] - \left(1-(1-\frac{\delta}{4})\beta_1\right)\sum_{t=k\tau+1}^{(k+1)\tau-1}\mathbb{E}\left[\left\|\widetilde{M}_l^{(t-1)} - \nabla_l f(x^{(t-1)})\right\|_F^2\right]$$

$$
\begin{aligned}
&\leq \left(1-\frac{\delta}{2}\right)\beta_1 \sum_{t=k\tau+1}^{(k+1)\tau-1} \mathbb{E}\left[\left\|\nabla_l f(x^{(t)})\right\|_F^2\right] \\
&\quad + \frac{5(1-\beta_1)}{\beta_1\delta} \sum_{t=k\tau+1}^{(k+1)\tau-1} \mathbb{E}\left[\left\|\nabla_l f(x^{(t)}) - \nabla_l f(x^{(t-1)})\right\|_F^2\right] \\
&\quad + \frac{10\beta_1}{\delta} \sum_{r=1}^{\tau-1} r \sum_{i=1}^{r} \mathbb{E}\left[\left\|\nabla_l f(x^{(k\tau+i)}) - \nabla_l f(x^{(k\tau+i-1)})\right\|_F^2\right] \\
&\quad + \beta_1^2\sigma_l^2(\tau-1) \\
&\leq \left(1-\frac{\delta}{2}\right)\beta_1 \sum_{t=k\tau+1}^{(k+1)\tau-1} \mathbb{E}\left[\left\|\nabla_l f(x^{(t)})\right\|_F^2\right] \\
&\quad + \left[\frac{5(1-\beta_1)}{\beta_1\delta} + \frac{5\beta_1\tau(\tau-1)}{\delta}\right] \sum_{t=k\tau}^{(k+1)\tau-2} \mathbb{E}\left[\left\|\nabla_l f(x^{(t+1)}) - \nabla_l f(x^{(t)})\right\|_F^2\right] \\
&\quad + \beta_1^2\sigma_l^2(\tau-1)
\end{aligned} \tag{6}
$$

Then add Eq. (6) and **Part 2** of Lemma A.3 together, we have

$$
\begin{aligned}
&\sum_{t=k\tau}^{(k+1)\tau-1} \mathbb{E}\left[\left\|\widetilde{M}_l^{(t)} - \nabla_l f(x^{(t)})\right\|_F^2\right] - \left(1-(1-\frac{\delta}{4})\beta_1\right) \sum_{t=k\tau}^{(k+1)\tau-1} \mathbb{E}\left[\left\|\widetilde{M}_l^{(t-1)} - \nabla_l f(x^{(t-1)})\right\|_F^2\right] \\
&\leq \left[\left(1-\frac{\delta}{2}\right)\beta_1 + \frac{2(1-\delta)}{\tau}\right] \sum_{t=k\tau}^{(k+1)\tau-1} \mathbb{E}\left[\left\|\nabla_l f(x^{(t)})\right\|_F^2\right] \\
&\quad + \left[\frac{5(1-\beta_1)}{\beta_1\delta} + \frac{5\beta_1\tau(\tau-1)}{\delta} + (\tau-1)(1-\delta)\right] \sum_{t=k\tau}^{(k+1)\tau-2} \mathbb{E}\left[\left\|\nabla_l f(x^{(t+1)}) - \nabla_l f(x^{(t)})\right\|_F^2\right] \\
&\quad + \beta_1^2\sigma_l^2\tau
\end{aligned} \tag{7}
$$

Then applying summation over k from 0 to K and summation over all $l \in [N]$ gives us the desired result. □

So far, we have the comparable result of upper bound of momentum error, in the next step, apply the same proof procedure as in [12] give us the convergence of low-rank MSGD with SARA.

Theorem A.5. *Under Assumptions 1–3, if hyperparameters*

$$
0 < \beta_1 \leq 1, \quad \tau \geq \frac{64}{3\beta_1\delta}, \quad 0 < \eta \leq \min\left\{\frac{1}{4L}, \sqrt{\frac{3\delta\beta_1^2}{80L^2}}, \sqrt{\frac{3\delta}{80\tau^2L^2}}, \sqrt{\frac{3\beta_1}{16\tau L^2}}\right\},
$$

MSGD-SARA with momentum re-projection converges as

$$
\frac{1}{K\tau}\sum_{t=0}^{K\tau-1} \mathbb{E}\left[\|\nabla f(x^{(t)})\|_2^2\right] \leq \frac{16\Delta}{\delta\eta K\tau} + \frac{32\beta_1\sigma^2}{3\delta}
$$

for any $K \geq 1$, where $\Delta = f(x^{(0)}) - \inf_x f(x)$.

Corollary A.6 (Convergence complexity of Low-rank MSGD with SARA)**.** *Under Assumption 3.1-3.2, if $T \geq 2 + 128/(3\delta) + (128\sigma)^2/(9\sqrt{\delta}L\Delta)$ and we choose*

$$
\beta_1 = \left(1+\sqrt{\frac{\delta^{3/2}\sigma^2T}{L\Delta}}\right)^{-1}, \qquad \tau = \left\lceil\frac{64}{3\delta\beta_1}\right\rceil, \qquad \eta = \left(4L + \sqrt{\frac{80L^2}{3\delta\beta_1^2} + \frac{80\tau^2L^2}{3\delta}} + \sqrt{\frac{16\tau L^2}{3\beta_1}}\right)^{-1},
$$

low-rank MSGD-SARA with momentum re-projection converges as

$$
\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\left[\|\nabla f(x^{(t)})\|_2^2\right] = \mathcal{O}\left(\frac{L\Delta}{\delta^{2.5}T} + \sqrt{\frac{L\Delta\sigma^2}{\delta^{3.5}T}}\right),
$$

where $\Delta = f(x^{(0)}) - \inf_x f(x)$.

B Experiment Implementation

To enable re-implementation, we provide hyperparameters herein. For fair comparison, we adopt the same hyperparameters for dominant subspace selection and SARA. Hyperparameters for GaLore with dominant subspace selection and SARA are shown in Table 5. Hyperparameters for Fira with dominant subspace selection and SARA are shown in Table 6.

Table 5: Hyperparameters for experiments with GaLore

Name	Values
Batch Size	512
Maximum Sequence Length	512
Warmup Steps	1000 for 60m, 2000 for 130m, 6000 for 350m, 10000 for 1.1B
Rank	128 for 60m, 256 for 130m, 256 for 350m, 512 for 1.1B
Weight Decay	0
Learning Rate	0.01
Scheduler	Cosine
Optimizer Specific Parameters	Adam: $\beta_1 = 0.9$, $\beta_2 = 0.999$ Adafactor: $\beta_1 = 0.9$, $\beta_2(t) = 1 - t^{-0.8}$ Adam-mini: $\beta_1 = 0.9$, $\beta_2 = 0.95$

Table 6: Hyperparameters for experiments with Fira

Name	Values
batch Size	512
Maximum sequence length	512
Warmup Steps	1000 for 60m, 2000 for 130m, 6000 for 350m
rank	128 for 60m, 256 for 130m, 256 for 350m
Weight Decay	0
Learning Rate	0.01
Scheduler	Cosine
optimizer specific parameters	Adam: $\beta_1 = 0.9, \beta_2 = 0.999$

C Limitations

Our work does not have any noteworthy limitations, as we directly address the key limitation of dominant subspace selection in prior work without requiring extra assumptions. In our paper, we provide a theoretical analysis of the convergence rate, though it still relies on the assumptions of L-smoothness (Assumption 3.1) and bounded, centered mini-batch gradient noise (Assumption 3.2).

However, we note that these assumptions are standard in the optimization literature.

D Societal Impact

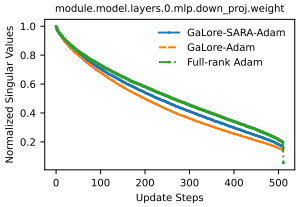
Regarding the positive societal impact, by reducing memory requirements for training, our SARA method enables more organizations—including those with limited compute budgets—to train or fine-tune competitive LLMs.

To the best of our knowledge, we do not anticipate any negative societal impacts.

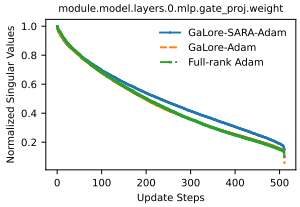
E More experimental results

E.1 High-Rank Updates

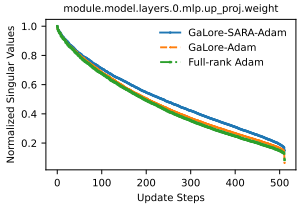
Now, we provide more experimental results for high-rank updates.



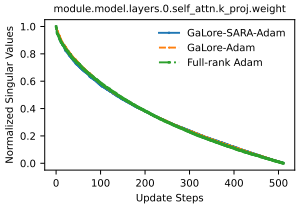
(a) Normalized singular values of $mlp.down\_proj$.



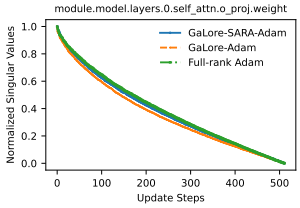
(b) Normalized singular values of $mlp.gate\_proj$.



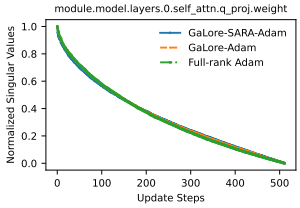
(c) Normalized singular values of $mlp.up\_proj$.



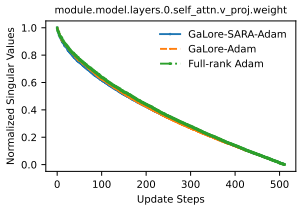
(d) Normalized singular values of $self\_attn.k\_proj$.



(e) Normalized singular values of $self\_attn.o\_proj$.

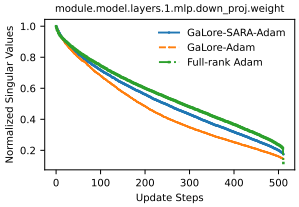


(f) Normalized singular values of $self\_attn.q\_proj$.

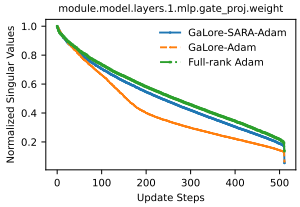


(g) Normalized singular values of $self\_attn.v\_proj$.

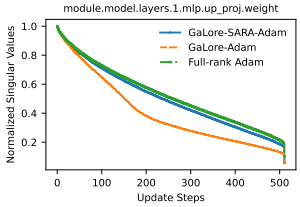
Figure 5: Normalized singular values of the weight difference between the $28k$-step checkpoint and $30k$-step checkpoint in different layers of Block 0 of LLaMA-$60M$ model during pretraining.



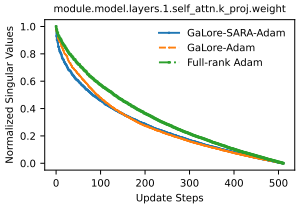
(a) Normalized singular values of $mlp.down\_proj$.



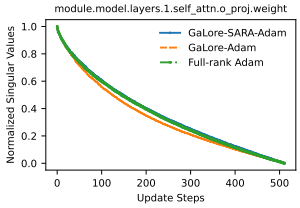
(b) Normalized singular values of $mlp.gate\_proj$.



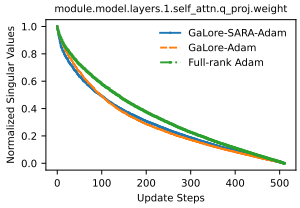
(c) Normalized singular values of $mlp.up\_proj$.



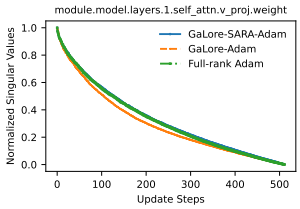
(d) Normalized singular values of $self\_attn.k\_proj$.



(e) Normalized singular values of $self\_attn.o\_proj$.

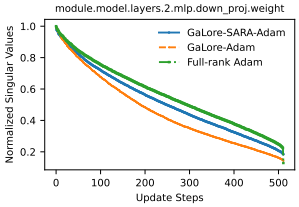


(f) Normalized singular values of $self\_attn.q\_proj$.

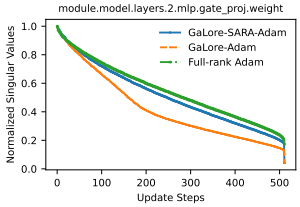


(g) Normalized singular values of $self\_attn.v\_proj$.

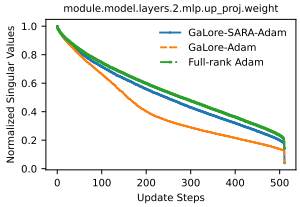
Figure 6: Normalized singular values of the weight difference between the $28k$-step checkpoint and $30k$-step checkpoint in different layers of Block 1 of LLaMA-$60M$ model during pretraining.



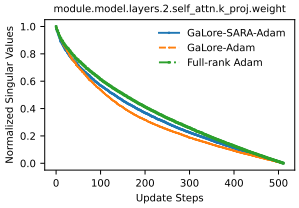
(a) Normalized singular values of $mlp.down\_proj$.



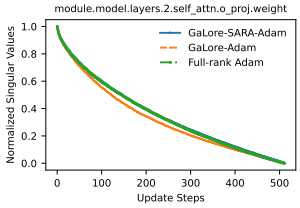
(b) Normalized singular values of $mlp.gate\_proj$.



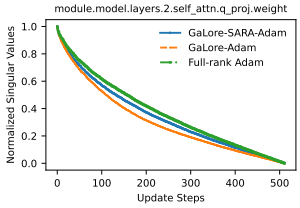
(c) Normalized singular values of $mlp.up\_proj$.



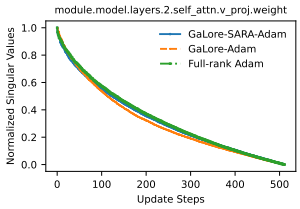
(d) Normalized singular values of $self\_attn.k\_proj$.



(e) Normalized singular values of $self\_attn.o\_proj$.

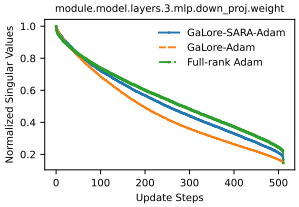


(f) Normalized singular values of $self\_attn.q\_proj$.

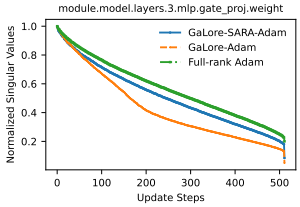


(g) Normalized singular values of $self\_attn.v\_proj$.

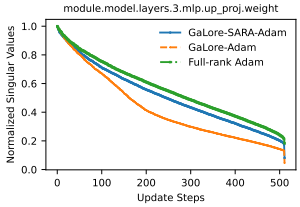
Figure 7: Normalized singular values of the weight difference between the $28k$-step checkpoint and $30k$-step checkpoint in different layers of Block 2 of LLaMA-$60M$ model during pretraining.



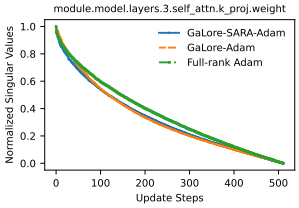
(a) Normalized singular values of $mlp.down\_proj$.



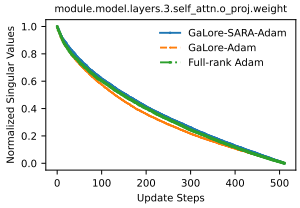
(b) Normalized singular values of $mlp.gate\_proj$.



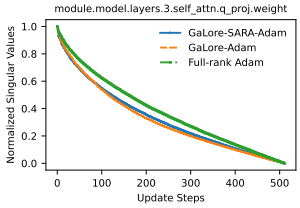
(c) Normalized singular values of $mlp.up\_proj$.



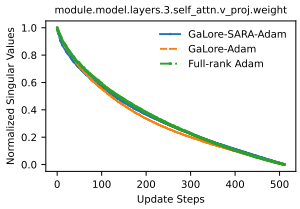
(d) Normalized singular values of $self\_attn.k\_proj$.



(e) Normalized singular values of $self\_attn.o\_proj$.

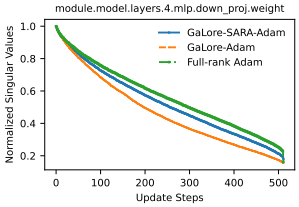


(f) Normalized singular values of $self\_attn.q\_proj$.

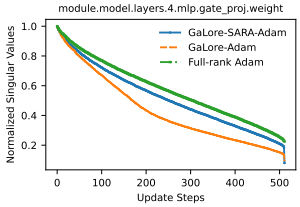


(g) Normalized singular values of $self\_attn.v\_proj$.

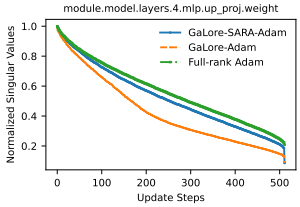
Figure 8: Normalized singular values of the weight difference between the $28k$-step checkpoint and $30k$-step checkpoint in different layers of Block 3 of LLaMA-$60M$ model during pretraining.



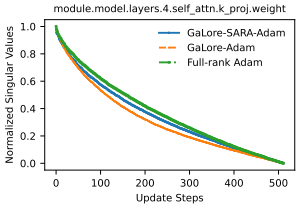
(a) Normalized singular values of $mlp.down\_proj$.



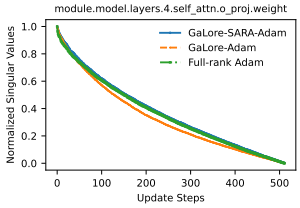
(b) Normalized singular values of $mlp.gate\_proj$.



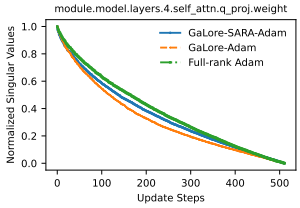
(c) Normalized singular values of $mlp.up\_proj$.



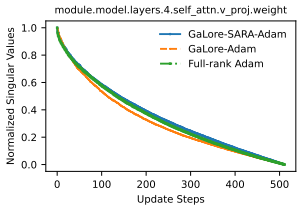
(d) Normalized singular values of $self\_attn.k\_proj$.



(e) Normalized singular values of $self\_attn.o\_proj$.

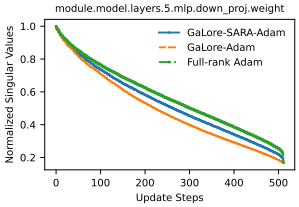


(f) Normalized singular values of $self\_attn.q\_proj$.

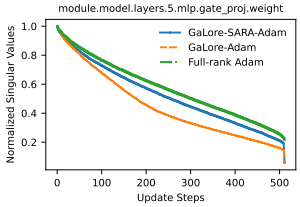


(g) Normalized singular values of $self\_attn.v\_proj$.

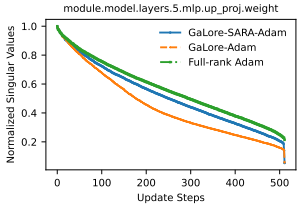
Figure 9: Normalized singular values of the weight difference between the $28k$-step checkpoint and $30k$-step checkpoint in different layers of Block 4 of LLaMA-$60M$ model during pretraining.



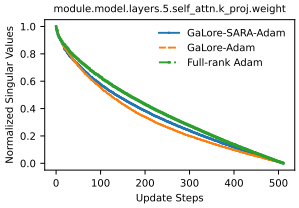
(a) Normalized singular values of $mlp.down\_proj$.



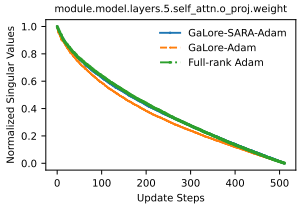
(b) Normalized singular values of $mlp.gate\_proj$.



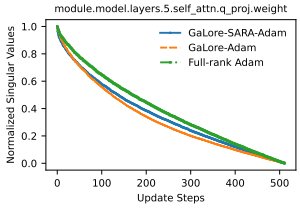
(c) Normalized singular values of $mlp.up\_proj$.



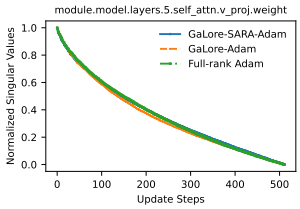
(d) Normalized singular values of $self\_attn.k\_proj$.



(e) Normalized singular values of $self\_attn.o\_proj$.

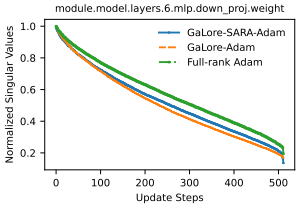


(f) Normalized singular values of $self\_attn.q\_proj$.

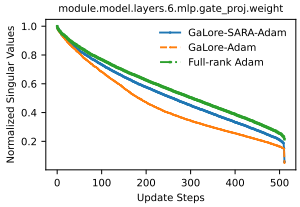


(g) Normalized singular values of $self\_attn.v\_proj$.

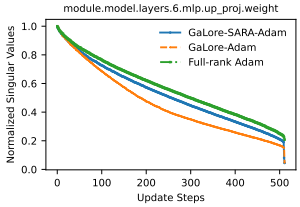
Figure 10: Normalized singular values of the weight difference between the $28k$-step checkpoint and $30k$-step checkpoint in different layers of Block 5 of LLaMA-$60M$ model during pretraining.



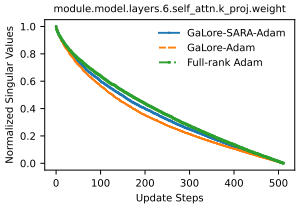
(a) Normalized singular values of $mlp.down\_proj$.



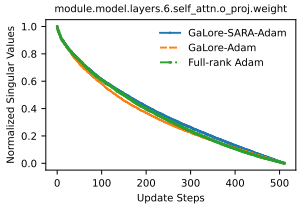
(b) Normalized singular values of $mlp.gate\_proj$.



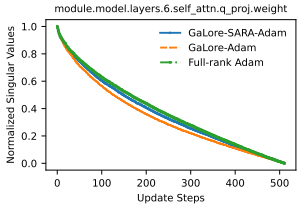
(c) Normalized singular values of $mlp.up\_proj$.



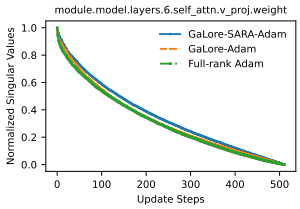
(d) Normalized singular values of $self\_attn.k\_proj$.



(e) Normalized singular values of $self\_attn.o\_proj$.

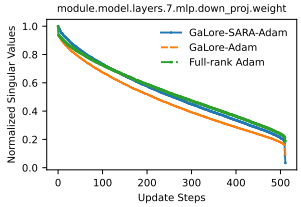


(f) Normalized singular values of $self\_attn.q\_proj$.

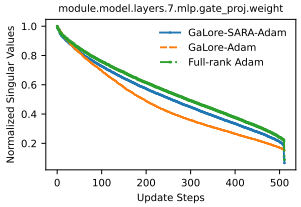


(g) Normalized singular values of $self\_attn.v\_proj$.

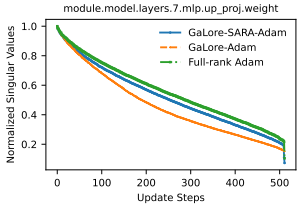
Figure 11: Normalized singular values of the weight difference between the $28k$-step checkpoint and $30k$-step checkpoint in different layers of Block 6 of LLaMA-$60M$ model during pretraining.



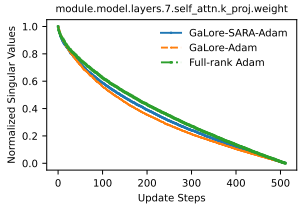
(a) Normalized singular values of $mlp.down\_proj$.



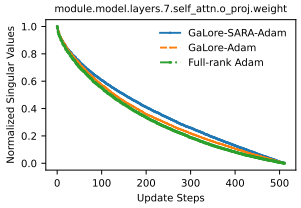
(b) Normalized singular values of $mlp.gate\_proj$.



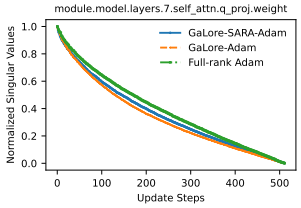
(c) Normalized singular values of $mlp.up\_proj$.



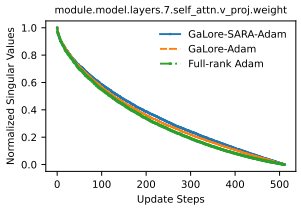
(d) Normalized singular values of $self\_attn.k\_proj$.



(e) Normalized singular values of $self\_attn.o\_proj$.



(f) Normalized singular values of $self\_attn.q\_proj$.

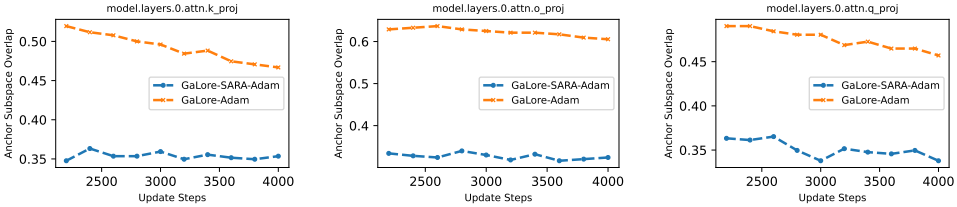


(g) Normalized singular values of $self\_attn.v\_proj$.

Figure 12: Normalized singular values of the weight difference between the $28k$-step checkpoint and $30k$-step checkpoint in different layers of Block 7 of LLaMA-$60M$ model during pretraining.

E.2 Anchor Similarity

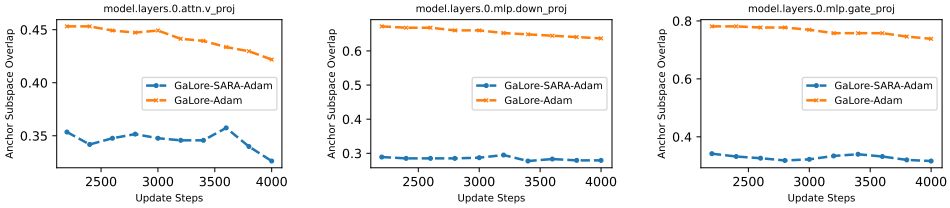
Now, we provide more experimental results for anchor similarity.



(a) Anchor subspace overlap of $attn.k\_proj$.

(b) Anchor subspace overlap of $attn.o\_proj$.

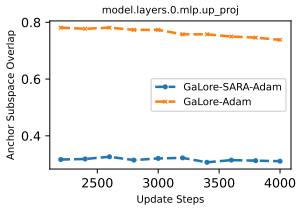
(c) Anchor subspace overlap of $attn.q\_proj$.



(d) Anchor subspace overlap of $attn.v\_proj$.

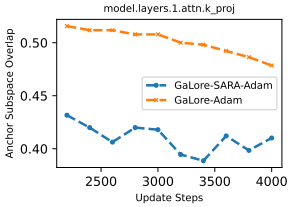
(e) Anchor subspace overlap of $mlp.down\_proj$.

(f) Anchor subspace overlap of $mlp.gate\_proj$.

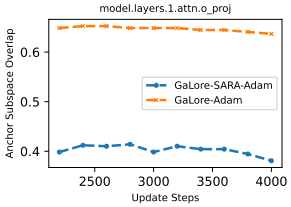


(g) Anchor subspace overlap of $mlp.up\_proj$.

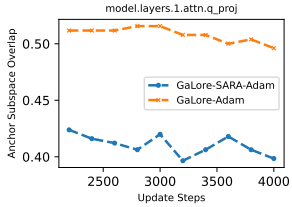
Figure 13: The low-rank subspace of different layers in Block 0 at the 2000-th iteration is taken as the anchor subspace. The figure shows the overlap between subspaces of the corresponding layer in Block 0 in later iterations and the anchor subspace.



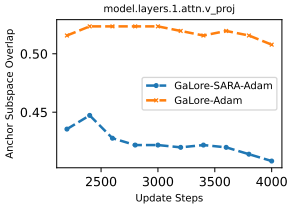
(a) Anchor subspace overlap of $attn.k\_proj$.



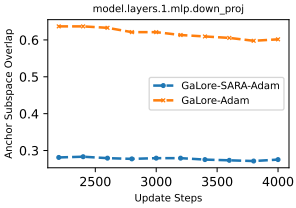
(b) Anchor subspace overlap of $attn.o\_proj$.



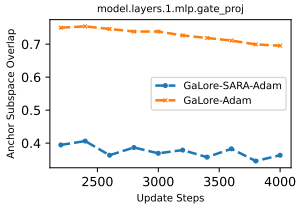
(c) Anchor subspace overlap of $attn.q\_proj$.



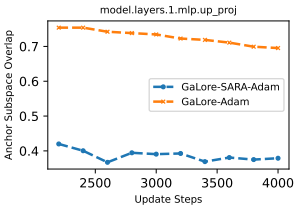
(d) Anchor subspace overlap of $attn.v\_proj$.



(e) Anchor subspace overlap of $mlp.down\_proj$.

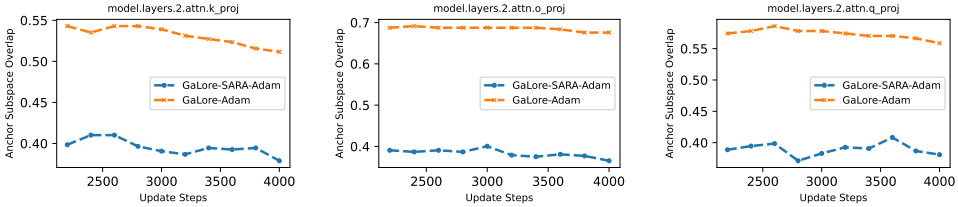


(f) Anchor subspace overlap of $mlp.gate\_proj$.



(g) Anchor subspace overlap of $mlp.up\_proj$.

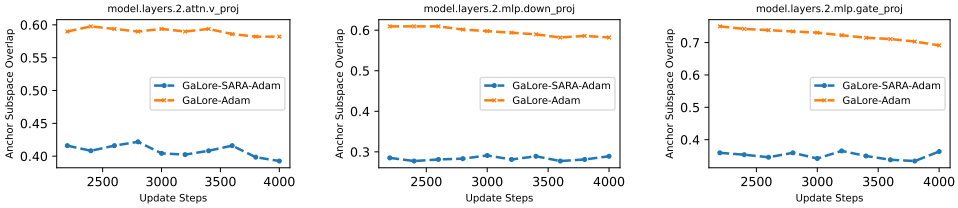
Figure 14: The low-rank subspace of different layers in Block 1 at the 2000-th iteration is taken as the anchor subspace. The figure shows the overlap between subspaces of the corresponding layer in Block 1 in later iterations and the anchor subspace.



(a) Anchor subspace overlap of $attn.k\_proj$.

(b) Anchor subspace overlap of $attn.o\_proj$.

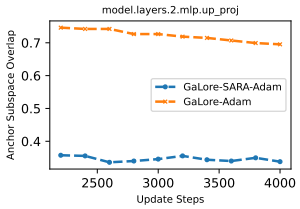
(c) Anchor subspace overlap of $attn.q\_proj$.



(d) Anchor subspace overlap of $attn.v\_proj$.

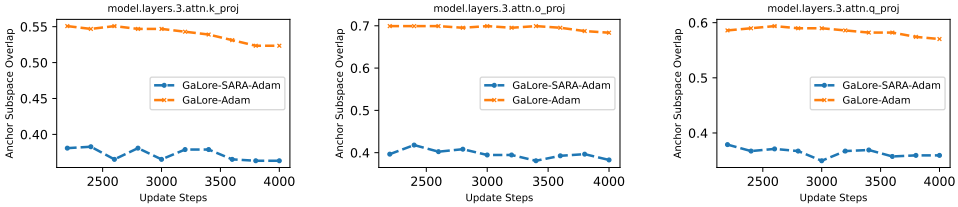
(e) Anchor subspace overlap of $mlp.down\_proj$.

(f) Anchor subspace overlap of $mlp.gate\_proj$.



(g) Anchor subspace overlap of $mlp.up\_proj$.

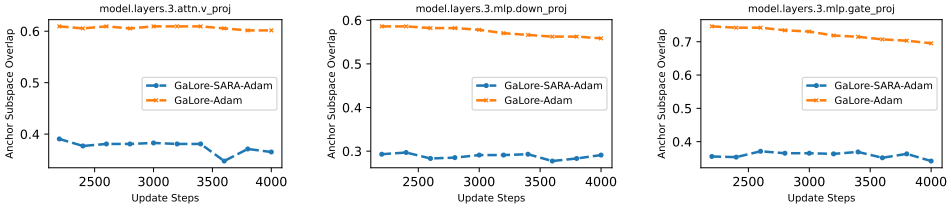
Figure 15: The low-rank subspace of different layers in Block 2 at the 2000-th iteration is taken as the anchor subspace. The figure shows the overlap between subspaces of the corresponding layer in Block 2 in later iterations and the anchor subspace.



(a) Anchor subspace overlap of $attn.k\_proj$.

(b) Anchor subspace overlap of $attn.o\_proj$.

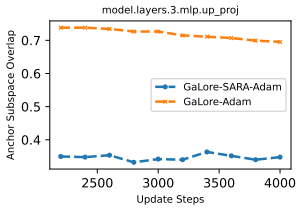
(c) Anchor subspace overlap of $attn.q\_proj$.



(d) Anchor subspace overlap of $attn.v\_proj$.

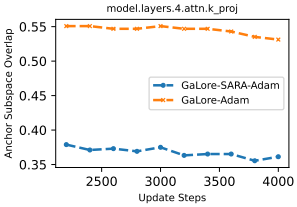
(e) Anchor subspace overlap of $mlp.down\_proj$.

(f) Anchor subspace overlap of $mlp.gate\_proj$.

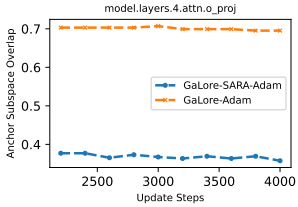


(g) Anchor subspace overlap of $mlp.up\_proj$.

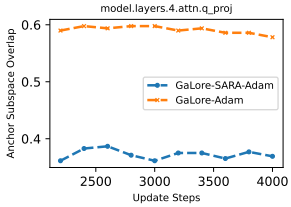
Figure 16: The low-rank subspace of different layers in Block 3 at the 2000-th iteration is taken as the anchor subspace. The figure shows the overlap between subspaces of the corresponding layer in Block 3 in later iterations and the anchor subspace.



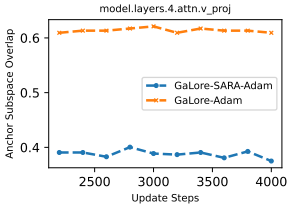
(a) Anchor subspace overlap of $attn.k\_proj$.



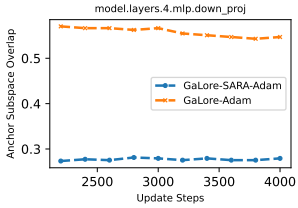
(b) Anchor subspace overlap of $attn.o\_proj$.



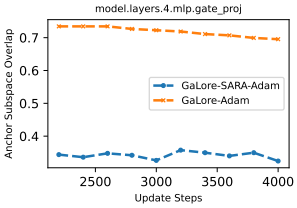
(c) Anchor subspace overlap of $attn.q\_proj$.



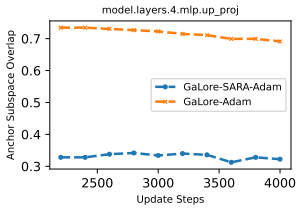
(d) Anchor subspace overlap of $attn.v\_proj$.



(e) Anchor subspace overlap of $mlp.down\_proj$.

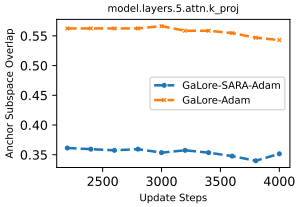


(f) Anchor subspace overlap of $mlp.gate\_proj$.

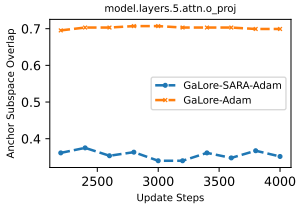


(g) Anchor subspace overlap of $mlp.up\_proj$.

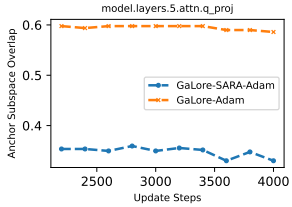
Figure 17: The low-rank subspace of different layers in Block 4 at the 2000-th iteration is taken as the anchor subspace. The figure shows the overlap between subspaces of the corresponding layer in Block 4 in later iterations and the anchor subspace.



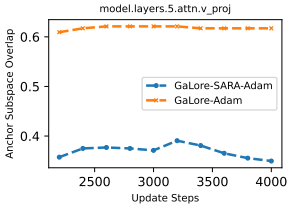
(a) Anchor subspace overlap of $attn.k\_proj$.



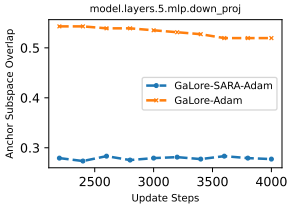
(b) Anchor subspace overlap of $attn.o\_proj$.



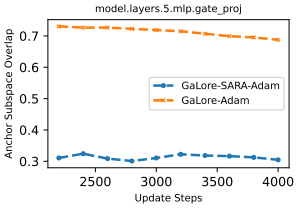
(c) Anchor subspace overlap of $attn.q\_proj$.



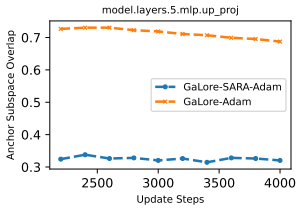
(d) Anchor subspace overlap of $attn.v\_proj$.



(e) Anchor subspace overlap of $mlp.down\_proj$.

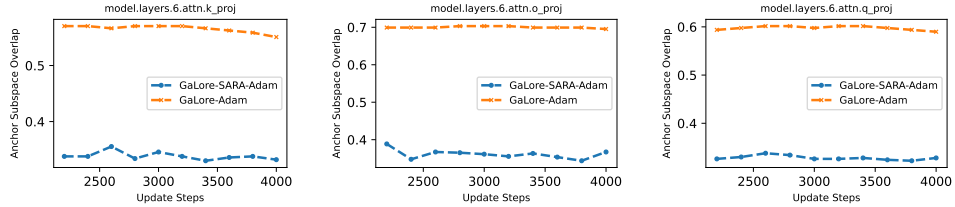


(f) Anchor subspace overlap of $mlp.gate\_proj$.



(g) Anchor subspace overlap of $mlp.up\_proj$.

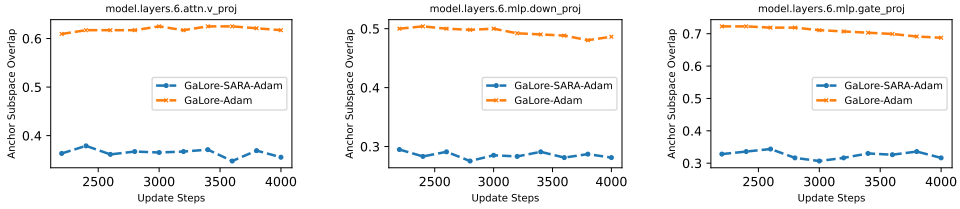
Figure 18: The low-rank subspace of different layers in Block 5 at the 2000-th iteration is taken as the anchor subspace. The figure shows the overlap between subspaces of the corresponding layer in Block 5 in later iterations and the anchor subspace.



(a) Anchor subspace overlap of $attn.k\_proj$.

(b) Anchor subspace overlap of $attn.o\_proj$.

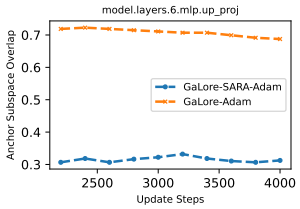
(c) Anchor subspace overlap of $attn.q\_proj$.



(d) Anchor subspace overlap of $attn.v\_proj$.

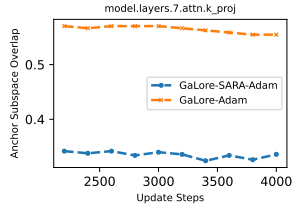
(e) Anchor subspace overlap of $mlp.down\_proj$.

(f) Anchor subspace overlap of $mlp.gate\_proj$.

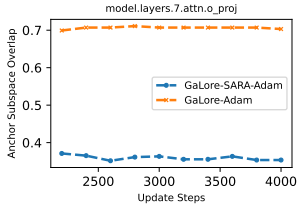


(g) Anchor subspace overlap of $mlp.up\_proj$.

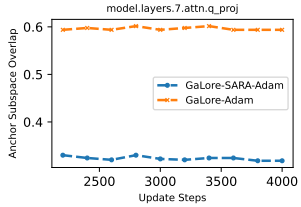
Figure 19: The low-rank subspace of different layers in Block 6 at the 2000-th iteration is taken as the anchor subspace. The figure shows the overlap between subspaces of the corresponding layer in Block 6 in later iterations and the anchor subspace.



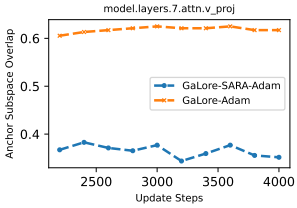
(a) Anchor subspace overlap of $attn.k\_proj$.



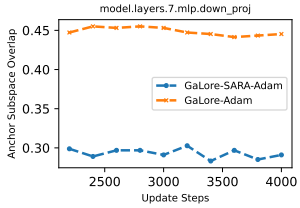
(b) Anchor subspace overlap of $attn.o\_proj$.



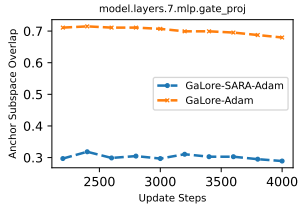
(c) Anchor subspace overlap of $attn.q\_proj$.



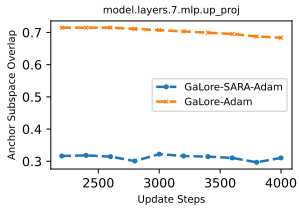
(d) Anchor subspace overlap of $attn.v\_proj$.



(e) Anchor subspace overlap of $mlp.down\_proj$.



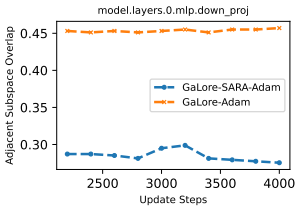
(f) Anchor subspace overlap of $mlp.gate\_proj$.

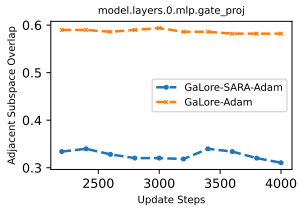


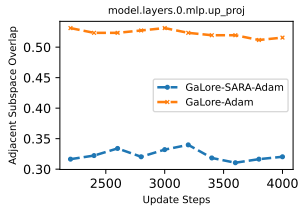
(g) Anchor subspace overlap of $mlp.up\_proj$.

Figure 20: The low-rank subspace of different layers in Block 7 at the 2000-th iteration is taken as the anchor subspace. The figure shows the overlap between subspaces of the corresponding layer in Block 7 in later iterations and the anchor subspace.

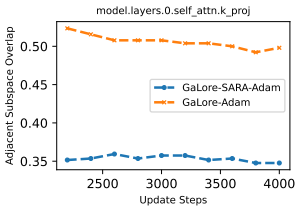
E.3 Adjacent Overlap

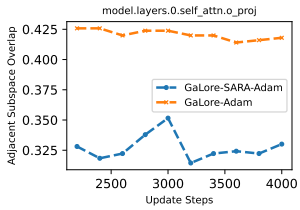


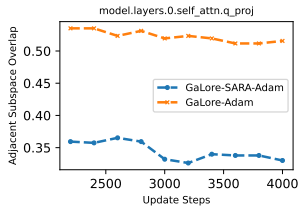




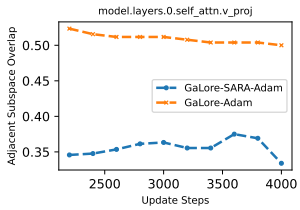
(a) Adjacent subspace overlap of $mlp.down\_proj$ (b) Adjacent subspace overlap of $mlp.gate\_proj$ (c) Adjacent subspace overlap of $mlp.up\_proj$





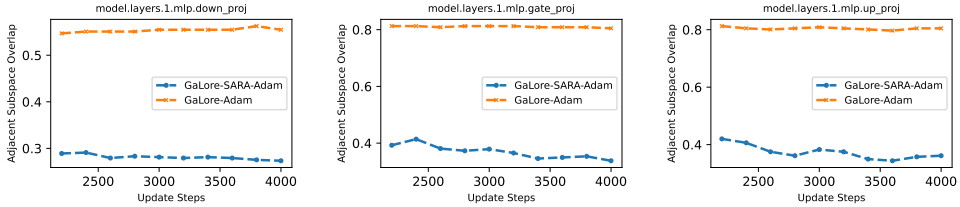


(d) Adjacent subspace overlap of $self\_attn.k\_proj$ (e) Adjacent subspace overlap of $self\_attn.o\_proj$ (f) Adjacent subspace overlap of $self\_attn.q\_proj$

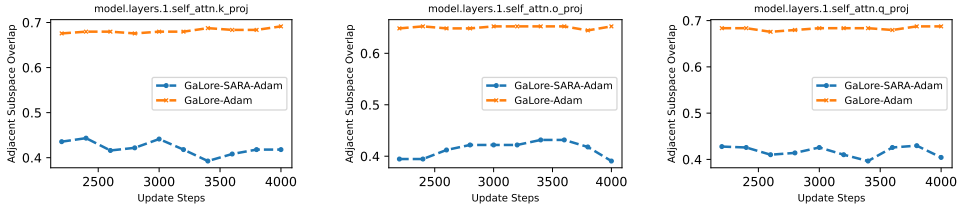


(g) Adjacent subspace overlap of $self\_attn.v\_proj$

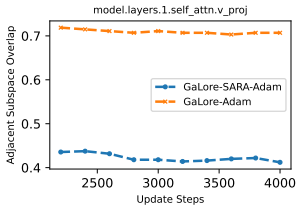
Figure 21: The overlap between adjacent subspaces of optimization trajectory of different layers in Block 0 in GaLore-Adam and GaLore-SARA-Adam during pretraining on the LLaMA-60M model between 2200-th and 4000-th iteration



(a) Adjacent subspace overlap of $mlp.down\_proj$ (b) Adjacent subspace overlap of $mlp.gate\_proj$ (c) Adjacent subspace overlap of $mlp.up\_proj$

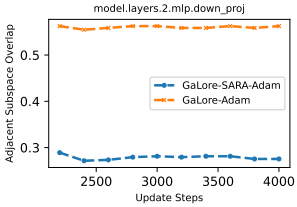


(d) Adjacent subspace overlap of $self\_attn.k\_proj$ (e) Adjacent subspace overlap of $self\_attn.o\_proj$ (f) Adjacent subspace overlap of $self\_attn.q\_proj$

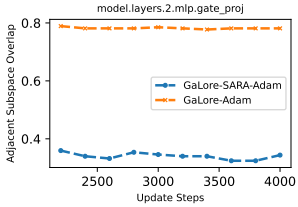


(g) Adjacent subspace overlap of $self\_attn.v\_proj$

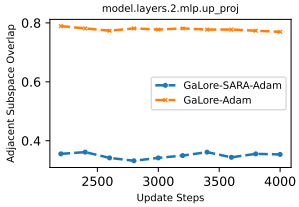
Figure 22: The overlap between adjacent subspaces of optimization trajectory of different layers in Block 1 in GaLore-Adam and GaLore-SARA-Adam during pretraining on the LLaMA-60M model between 2200-th and 4000-th iteration



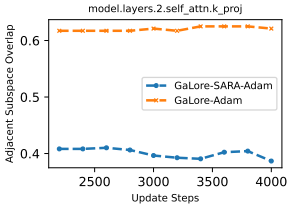
(a) Adjacent subspace overlap of $mlp.down\_proj$



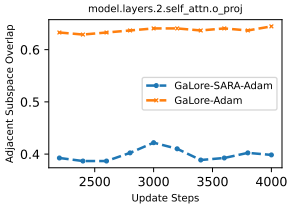
(b) Adjacent subspace overlap of $mlp.gate\_proj$



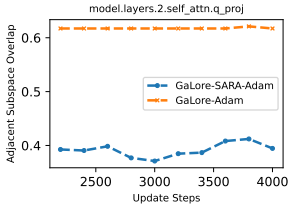
(c) Adjacent subspace overlap of $mlp.up\_proj$



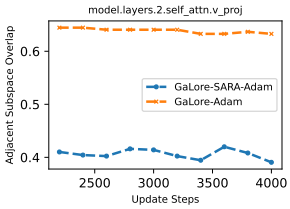
(d) Adjacent subspace overlap of $self\_attn.k\_proj$



(e) Adjacent subspace overlap of $self\_attn.o\_proj$

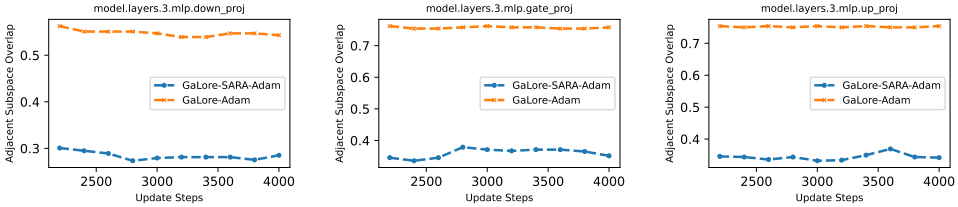


(f) Adjacent subspace overlap of $self\_attn.q\_proj$

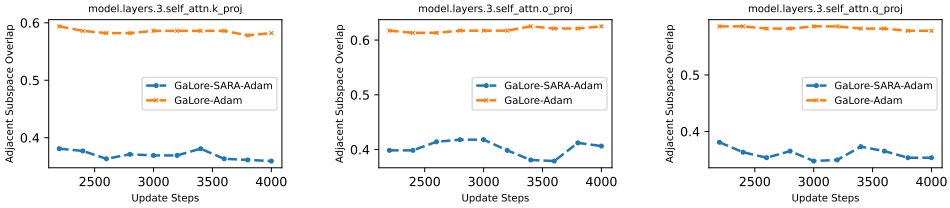


(g) Adjacent subspace overlap of $self\_attn.v\_proj$

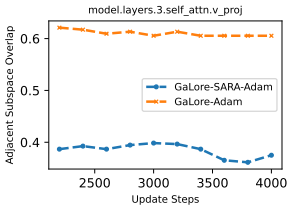
Figure 23: The overlap between adjacent subspaces of optimization trajectory of different layers in Block 2 in GaLore-Adam and GaLore-SARA-Adam during pretraining on the LLaMA-60M model between 2200-th and 4000-th iteration



(a) Adjacent subspace overlap of $mlp.down\_proj$ (b) Adjacent subspace overlap of $mlp.gate\_proj$ (c) Adjacent subspace overlap of $mlp.up\_proj$

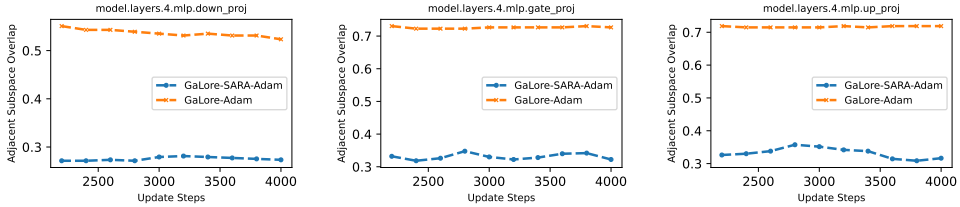


(d) Adjacent subspace overlap of $self\_attn.k\_proj$ (e) Adjacent subspace overlap of $self\_attn.o\_proj$ (f) Adjacent subspace overlap of $self\_attn.q\_proj$

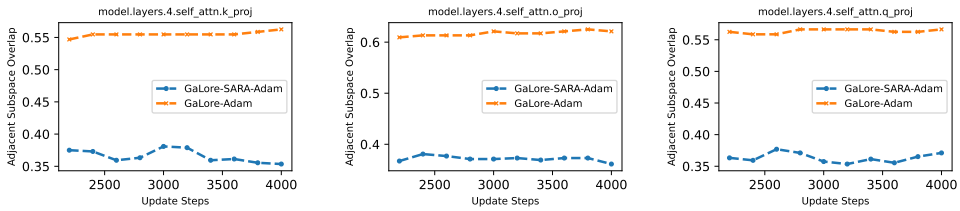


(g) Adjacent subspace overlap of $self\_attn.v\_proj$

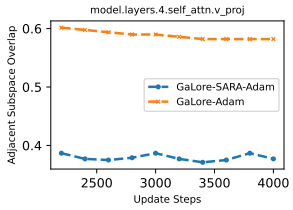
Figure 24: The overlap between adjacent subspaces of optimization trajectory of different layers in Block 3 in GaLore-Adam and GaLore-SARA-Adam during pretraining on the LLaMA-60M model between 2200-th and 4000-th iteration



(a) Adjacent subspace overlap of $mlp.down\_proj$ (b) Adjacent subspace overlap of $mlp.gate\_proj$ (c) Adjacent subspace overlap of $mlp.up\_proj$

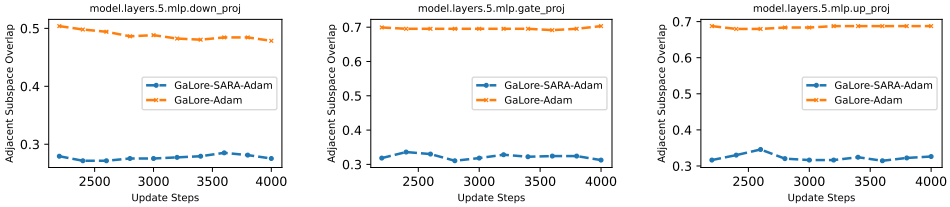


(d) Adjacent subspace overlap of $self\_attn.k\_proj$ (e) Adjacent subspace overlap of $self\_attn.o\_proj$ (f) Adjacent subspace overlap of $self\_attn.q\_proj$

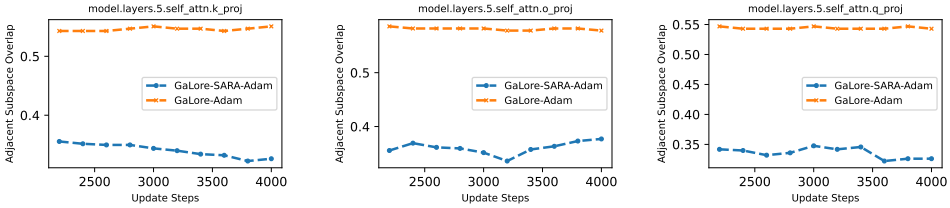


(g) Adjacent subspace overlap of $self\_attn.v\_proj$

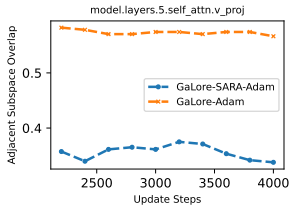
Figure 25: The overlap between adjacent subspaces of optimization trajectory of different layers in Block 4 in GaLore-Adam and GaLore-SARA-Adam during pretraining on the LLaMA-60M model between 2200-th and 4000-th iteration



(a) Adjacent subspace overlap of $mlp.down\_proj$ (b) Adjacent subspace overlap of $mlp.gate\_proj$ (c) Adjacent subspace overlap of $mlp.up\_proj$

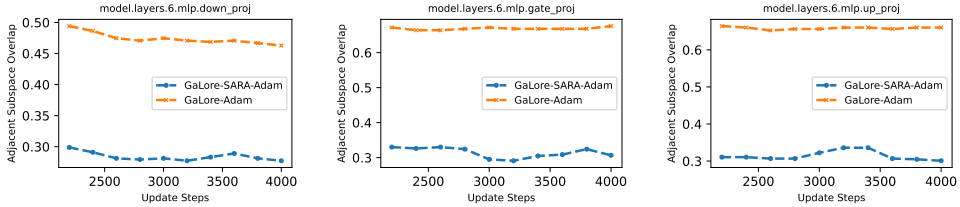


(d) Adjacent subspace overlap of $self\_attn.k\_proj$ (e) Adjacent subspace overlap of $self\_attn.o\_proj$ (f) Adjacent subspace overlap of $self\_attn.q\_proj$

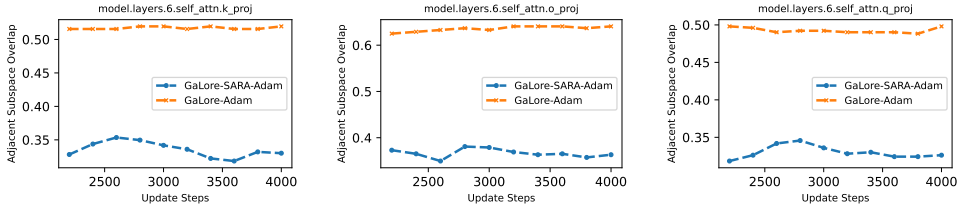


(g) Adjacent subspace overlap of $self\_attn.v\_proj$

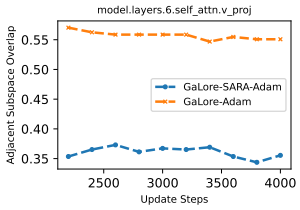
Figure 26: The overlap between adjacent subspaces of optimization trajectory of different layers in Block 5 in GaLore-Adam and GaLore-SARA-Adam during pretraining on the LLaMA-60M model between 2200-th and 4000-th iteration



(a) Adjacent subspace overlap of $mlp.down\_proj$ (b) Adjacent subspace overlap of $mlp.gate\_proj$ (c) Adjacent subspace overlap of $mlp.up\_proj$

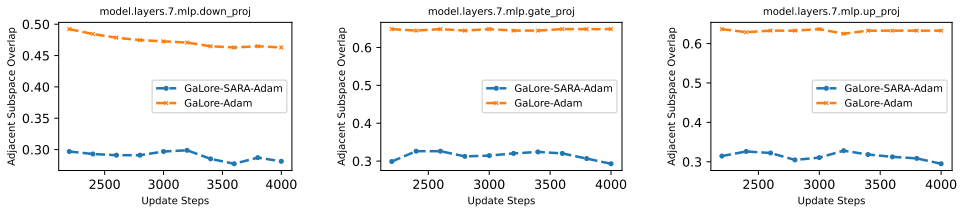


(d) Adjacent subspace overlap of $self\_attn.k\_proj$ (e) Adjacent subspace overlap of $self\_attn.o\_proj$ (f) Adjacent subspace overlap of $self\_attn.q\_proj$



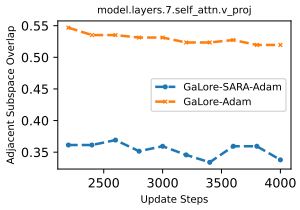
(g) Adjacent subspace overlap of $self\_attn.v\_proj$

Figure 27: The overlap between adjacent subspaces of optimization trajectory of different layers in Block 6 in GaLore-Adam and GaLore-SARA-Adam during pretraining on the LLaMA-60M model between 2200-th and 4000-th iteration



(a) Adjacent subspace overlap of $mlp.down\_proj$ (b) Adjacent subspace overlap of $mlp.gate\_proj$ (c) Adjacent subspace overlap of $mlp.up\_proj$

(d) Adjacent subspace overlap of $self\_attn.k\_proj$ (e) Adjacent subspace overlap of $self\_attn.o\_proj$ (f) Adjacent subspace overlap of $self\_attn.q\_proj$



(g) Adjacent subspace overlap of $self\_attn.v\_proj$

Figure 28: The overlap between adjacent subspaces of optimization trajectory of different layers in Block 7 in GaLore-Adam and GaLore-SARA-Adam during pretraining on the LLaMA-60M model between 2200-th and 4000-th iteration