GRASP: Replace Redundant Layers with Adaptive Singular Parameters for Efficient Model Compression

Kainan Liu[1,2,†], **Yong Zhang**[1,†], **Ning Cheng**[1,*], **Zhitao Li**[1],
Shaojun Wang[1], **Jing Xiao**[1],
[1]Ping An Technology (Shenzhen) Co., Ltd., China
[2]The Hong Kong University of Science and Technology (Guangzhou)
{zhangyong203, chengning211}@pingan.com.cn

Abstract

Recent studies have demonstrated that many layers are functionally redundant in large language models (LLMs), enabling model compression by removing these layers to reduce inference cost. While such approaches can improve efficiency, indiscriminate layer pruning often results in significant performance degradation. In this paper, we propose **GRASP** (Gradient-based Retention of Adaptive Singular Parameters), a novel compression framework that mitigates this issue by preserving sensitivity-aware singular values. Unlike direct layer pruning, GRASP leverages gradient-based attribution on a small calibration dataset to adaptively identify and retain critical singular components. By replacing redundant layers with only a minimal set of parameters, GRASP achieves efficient compression while maintaining strong performance with minimal overhead. Experiments across multiple LLMs show that GRASP consistently outperforms existing compression methods, achieving 90% of the original model's performance under 20% compression ratio. The source code is available at https://github.com/LyoAI/GRASP.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities across a wide range of tasks, including language generation, reasoning, and question answering (Brown et al., 2020; Touvron et al., 2023b). However, their massive parameter sizes pose computational and memory challenges, hindering deployment on resource-limited devices (Zhou et al., 2024). To address this, model compression techniques such as quantization (Frantar et al., 2022; Lin et al., 2024; Xiao et al., 2023), knowledge distillation (Gu et al., 2023; Xu et al., 2024), and pruning(Sun et al., 2024; Ashkboos et al., 2024) have been widely explored. Among these, structured pruning methods remove entire components such as neurons or layers to streamline the model, thereby achieving hardware efficiency and inference speedup.

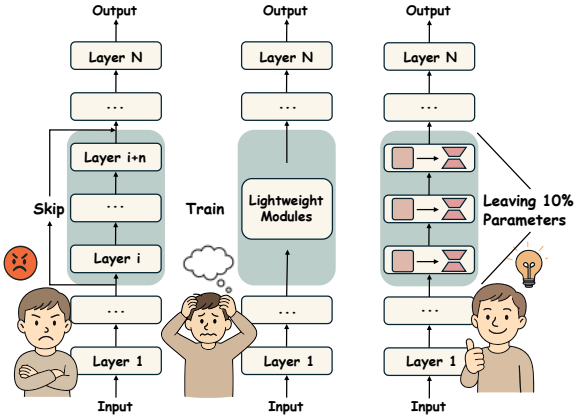

Figure 1: Unlike conventional layer pruning, which either skips redundant layers—often causing moderate performance drops—or replaces them with lightweight modules that require additional training, GRASP (right) retains only the most critical 10% of parameters within the redundant layers, effectively preserving accuracy with minimal overhead.

In this work, we focus on structured layer pruning, which builds on prior findings that certain consecutive layers in large language models (LLMs) are functionally redundant. These findings have inspired approaches that either remove such layers entirely (Men et al., 2024, Yang et al., 2024, Kim et al., 2024) or replace them with lightweight modules (Chen et al., 2024) to reduce inference cost. Although layer removal is simple and computationally efficient, it often results in significant performance degradation. This degradation arises from disrupted information flow and misaligned intermediate representations (Liang et al., 2024), suggesting that layer removal eliminates certain important components that contribute meaningfully to maintaining model performance. Replacing redun-

† Equal contribution.
∗ Corresponding author.
This work was done during Kainan Liu's internship at Ping An Technology.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 26333–26348
November 4-9, 2025

dant layers with lightweight modules can mitigate this issue to some extent, but such modules are typically randomly initialized, requiring substantial computational resources for training.

In this paper, we propose **GRASP** (Gradient-based Retention of Adaptive Singular Parameters), a novel compression framework that replaces redundant layers with adaptive singular parameters for efficient LLM compression. Unlike direct layer removal, GRASP exploits the low-rank structure inherent in redundant layers, replacing the redundant layers with only a small subset of parameters while maintaining strong model performance. Specifically, GRASP operates in two key steps: First, it identifies layers suitable for pruning based on the cosine similarity of output hidden states between adjacent layers. Then, instead of relying on magnitude-based heuristics, GRASP leverages gradient attribution derived from a small calibration dataset to adaptively identify and retain the singular values most critical for downstream task performance.

To evaluate the effectiveness of GRASP, We conduct extensive experiments on 19 datasets and 5 models from two distinct LLM families (LLaMA and Mistral). Notably, GRASP achieves strong performance in a training-free setting, requiring no additional optimization. Furthermore, when post-training compensation is applied, only a small number of samples are needed to rapidly restore model performance. This efficiency arises from retaining critical components within redundant layers, rather than relying on randomly initialized replacements. Overall, this paper makes the following contributions:

- We propose **GRASP**, a novel training-free compression framework that replaces redundant layers with adaptive singular parameters, leveraging the low-rank structure within LLMs to preserve performance with minimal overhead.
- We introduce a gradient-based singular value selection mechanism, enabling efficient identification of critical components without relying on magnitude-based heuristics.
- We conduct extensive experiments across ten datasets and five models from two major LLM families (LLaMA and Mistral), demonstrating that GRASP consistently achieves strong performance under both training-free and low-resource fine-tuning settings.

2 Method

Figure 1 illustrates the workflow of GRASP. The method consists of two main steps: Identifying redundant layers to be pruned (Sec 2.1) and replacing redundant layers with critical singular components guided by gradient-based attribution (Sec 2.2). Below, we describe each step in detail.

2.1 Redundant Layer Selection

The first step in GRASP is identifying redundant layers. These are layers that contribute minimally to the transformation of hidden states, exhibiting high redundancy and limited impact on overall model performance. Following prior works (Song et al., 2024; Chen et al., 2024), we use cosine similarity to quantify the degree of transformation in a given layer.

For a transformer layer with input hidden state $H_i \in \mathbb{R}^d$ and output hidden state $H_{i+1} \in \mathbb{R}^d$, the cosine similarity is computed as:

$$\cos(H_i, H_{i+1}) = \frac{H_i^T H_{i+1}}{\|H_i\|_2 \|H_{i+1}\|_2} \quad (1)$$

A high cosine similarity indicates minimal transformation, suggesting that the layer is redundant. Instead of directly removing these layers, GRASP compresses weight matrices with a gradient-based approach to retain critical internal transformations.

2.2 Layer Replacement with Adaptive Singular Parameters

Motivation. GRASP operates under the hypothesis that redundant layers exhibit an inherent low-rank structure, allowing their functionality to be effectively approximated using low-rank matrices. Based on this insight, a straightforward approach is to apply singular value decomposition (SVD) to these layers. However, prioritizing components solely based on singular value magnitude does not necessarily correlate with downstream task performance (Hsu et al., 2022; Hua et al., 2025). To validate this point, we selectively zero out groups of singular values in the weight matrices of large language models and measure their impact on downstream tasks. As shown in Figure 2, we make two key observations: (1) The contribution of a singular value to downstream task performance is not solely determined by its magnitude; smaller values can be crucial for task performance in some cases. (2) Redundant layers exhibit a highly low-rank structure, where only a few singular directions dominate the model performance.

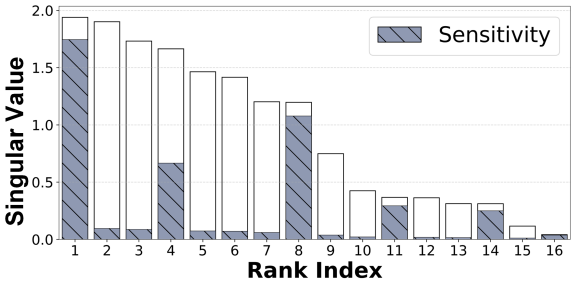

Figure 2: Sensitivity analysis of grouped singular value truncation. While singular values are typically ordered by magnitude, their impact on downstream performance does not follow the same order.

Key Design. To address this limitation, GRASP introduces gradient-based attribution to evaluate the importance of each singular value by its contribution to model performance, rather than relying on magnitude alone. Formally, the singular value decomposition of a weight matrix $W \in \mathbb{R}^{m\times n}$ is given by $W = U\Sigma V^\top$. Equivalently, W can be expressed as the sum of its rank-one components:

$$W = \sum_{k=1}^{l} u_k \sigma_k v_k^\top, \tag{2}$$

where $l = \min(m, n)$, σ_k denotes the k-th singular value, and u_k, v_k are the k-th column vectors of the orthogonal matrices U and V, respectively. For brevity, we use Φ_k to represent the singular group $\{u_k, \sigma_k, v_k^\top\}$. GRASP estimates the importance of each Φ_k using a small, general-purpose calibration dataset (e.g., WikiText-2), computing a sensitivity-based score that reflects its effect on model performance, given by:

$$I(\Phi_k) = T(\sigma_k) + \sum_{i=1}^{m} T(u_{k,i}) + \sum_{i=1}^{n} T(v_{k,i}) \tag{3}$$

where $T(\cdot)$ denotes the estimated loss change when a certain parameter θ is zeroed out. This can be approximated by the second-order Taylor expansion (LeCun et al., 1989), defined as:

$$T(\theta) = \left|\theta^\top \nabla_\theta \mathcal{L} + \frac{1}{2}\theta^\top H \theta + \mathcal{O}(\|\theta\|^3)\right| \tag{4}$$

$$\mathcal{L} = -\sum_t \log P(y_t \mid x_{\le t}) \tag{5}$$

where $\nabla_\theta \mathcal{L}$ denotes the gradient of the standard language modeling objective function $\mathcal{L}$ w.r.t. the parameter θ, and H represents the corresponding Hessian matrix. To reduce computational overhead, we omit the second-order term and approximate the importance using only the first-order derivative (Hua et al., 2025, Kim et al., 2024):

$$I(\Phi_k) = \left|\sigma_k \frac{\partial \mathcal{L}}{\partial \sigma_k}\right| + \sum_{i=1}^{m}\left|u_{k,i}\frac{\partial \mathcal{L}}{\partial u_{k,i}}\right| + \sum_{i=1}^{n}\left|v_{k,i}\frac{\partial \mathcal{L}}{\partial v_{k,i}}\right| \tag{6}$$

The first term in Eq. 6 captures the gradient projection of the loss $\mathcal{L}$ w.r.t. the singular value σ_i. The proof is given in AppendixA.1. By aggregating the first-order expansions across all components within each singular group, our method effectively captures the contribution of singular values, where larger values indicate greater importance. This gradient-oriented attribution moves beyond heuristic magnitude-based criteria, enabling performance-aligned importance evaluation. Under the hypothesis that redundant layers exhibit an inherent low-rank structure, we retain only the top-$r\%$ singular values most critical to model performance based on Eq. 6, and use them to replace the corresponding redundant layer. The relationship between the singular group retain ratio ($r\%$) and the overall target compression ratio is detailed in Appendix A.3.

2.3 Detailed Implementation of GRASP

GRASP processes the redundant layers in a sequential manner, starting from the final redundant layer and proceeding backward through the network. We summarize the detailed algorithm in Algorithm 1.

Algorithm 1 GRASP: Gradient-based Retention of Adaptive Singular Parameters

Require: Model M, Calibration set D, Retain ratio r
Ensure: Compressed model $\tilde{M}$

Step 1: Redundant Layer Selection:
Compute cosine similarity $\cos(H_i, H_{i+1})$ for all layers via Eq.1 using D
Select top-L layers with highest similarity as redundant
Step 2: Gradient-Guided Compression:
for each redundant layer l (in reverse order) **do**
  for each $W \in$ {attention, MLP} **do**
    SVD: $W = U\Sigma V^\top$
    Compute importance $I(\Phi_k)$ for each $\Phi_k = \{u_k, \sigma_k, v_k^\top\}$ via Eq.6 using D
    Keep top-$r\%$ singular groups, reconstruct $\tilde{W}$
  end for
end for
return $\tilde{M}$

3 Experiments

In this section, we conduct comprehensive experiments to evaluate GRASP from three key perspectives. (1) We first compare our method with existing pruning-based LLM compression approaches to demonstrate its effectiveness (Section 3.2). (2) Next, we analyze the inference speed-up achieved by GRASP (Section 3.3). (3) Finally, we investigate the factors influencing our approach by performing ablation studies on the choice of calibration datasets and pruning strategies. (Section 3.4)

3.1 Experimental Setup

Below we detail the models, benchmarks, baselines and implementation details used in our experiments, with more experimental setups provided in Appendix A.2.

Models. We evaluate GRASP on a range of large language models (LLMs) from two model families: the LLaMA family, including LLaMA-7B (Touvron et al., 2023a), LLaMA 2-7B, LLaMA 2-13B (Touvron et al., 2023b), and LLaMA 3.1-8B-Instruct (Dubey et al., 2024), as well as Mistral-7B (Jiang et al., 2023) from the Mistral family.

Baselines. We compare GRASP against 8 structured pruning methods to substantiate its efficacy:

- **Layer-pruning methods** We consider three representative layer-pruning methods as baselines: ShortGPT (Men et al., 2024), LaCo (Yang et al., 2024) and LLM-Streamline (Chen et al., 2024).
- **Module-pruning methods** We also select LLM-Pruner (Ma et al., 2023) and SliceGPT (Ashkboos et al., 2024) which prune the redundant modules in LLMs.
- **Low-rank Pruning methods** Considering our method involves Gradient-based SVD, we also compare with other low-rank pruning methods: FWSVD (Hsu et al., 2022), ASVD (Yuan et al., 2023) and SVD-LLM (Wang et al., 2024).

We provide a detailed comparison of these pruning-based LLM compression methods in Appendix A.4.

Implementation Details. To ensure a fair comparison, we randomly sample 512 data points from the **WikiText-2** dataset as the calibration dataset. All experiments are conducted on NVIDIA A100-SXM4 (80GB) GPUs. Further experimental details can be found in Appendix A.2.

3.2 Comparison with Pruning-based LLM Compression Methods

3.2.1 Comparison without Post-Training Compensation

Models and Benchmarks. In this experiment, we evaluate GRASP against representative structured pruning baselines using a more modern LLM architecture—LLaMA 3.1-8B-Instruct—**without applying any post-training compensation**. The model is compressed to 20% of its original size and evaluated on seven commonsense reasoning benchmarks, including **WinoGrande** (Sakaguchi et al., 2020), **HellaSwag** (Zellers et al., 2019), **OpenBookQA** (Mihaylov et al., 2018), **PIQA** (Bisk et al., 2020), **ARC-e**, **ARC-c** (Clark et al., 2018), and **MathQA** (Amini et al., 2019). All tasks are tested in a zero-shot setting using the LM-Evaluation-Harness framework (Gao et al., 2024).

Main Results. As shown in Table 1, GRASP achieves the highest average accuracy across seven commonsense reasoning benchmarks, consistently outperforming all baseline methods. In particular, GRASP improves over SliceGPT by 34% in average accuracy and outperforms LaCo by 12%. To further evaluate the generalizability of GRASP across different LLM architectures, we additionally conduct experiments on LLaMA 2-7B, LLaMA 2-13B, and Mistral-7B. Detailed results are presented in Appendix A.5. Notably, GRASP demonstrates significantly improved robustness across models, effectively mitigating the variability in pruning sensitivity across diverse architectures.

3.2.2 Comparison with Post-Training Compensation

Models and Benchmarks. In this section, following prior research, we compress the LLaMA 2-7B model under a 25% compression ratio and evaluate the compressed model on a broad set of natural language understanding (NLU) and question-answering (QA) benchmarks, including **CMNLI** (Xu et al., 2020), **HellaSwag** (Zellers et al., 2019), **PIQA** (Bisk et al., 2020), **CHID** (Zheng et al., 2019), **WSC** (Levesque et al., 2012), **CommonsenseQA** (Talmor et al., 2018), **BoolQ** (Clark et al., 2019), **MMLU** (Hendrycks et al., 2020), **CMMLU** (Li et al., 2023), **Race** (Lai et al., 2017) and **C3** (Sun et al., 2020). We utilized the OpenCompass evaluation framework (Contributors, 2023) and report accuracy as the evaluation metric for all benchmarks under the PPL mode, fol-

Methods	OpenbookQA	ARC_e	WinoGrande	HellaSwag	ARC_c	PIQA	MathQA	Average	Percentage
Dense	0.34	0.82	0.74	0.59	0.52	0.80	0.39	0.60	100.0%
LaCo	**0.26**	0.49	0.65	0.33	0.30	0.65	**0.30**	0.42	70.9%
ShortGPT	0.21	0.57	0.66	0.42	0.32	0.67	0.26	0.44	74.1%
SliceGPT	0.15	0.43	0.51	0.30	0.23	0.58	0.22	0.35	57.7%
GRASP	0.22	**0.60**	**0.70**	**0.44**	**0.37**	**0.69**	0.28	**0.47**	**78.6%**

Table 1: Zero-shot performance of GRASP and structured pruning baselines without post-training under a 20% compression ratio. Results are reported on seven reasoning datasets (individual and average accuracy). Bold values indicate the best performance.

Method	C3	CMNLI	CHID	BoolQ	WSC	CoQA	HeSW	PIQA	Race-M	Race-H	MMLU	CMMLU	Avg.	Per.
Dense	43.8	33.0	41.6	70.8	37.5	66.7	71.3	78.1	33.1	35.5	46.8	31.8	49.2	100.0%
LLMPruner*	29.7	33.4	**28.4**	58.7	40.4	48.5	54.6	72.0	22.9	22	25.3	25.0	38.4	78.1%
SliceGPT*	31.5	31.6	18.5	59.9	**43.3**	49.6	47.5	68.3	27.0	29.4	28.8	24.8	38.4	78.0%
LaCo*	39.7	34.4	36.1	64.1	40.4	45.7	55.7	69.8	23.6	22.6	26.5	25.2	40.3	82.0%
ShortGPT*	40.2	34.4	21.5	67.3	40.4	51.7	59.7	69.0	35.2	34.7	44.6	28.9	44.0	89.4%
LLM-Streamline-FFN*	40.7	33.0	22.8	65.9	38.5	60.6	61.2	71.2	**38.0**	**38.7**	**47.0**	**31.7**	45.8	93.1%
LLM-Streamline-Layer*	43.3	33.0	24.1	67.5	36.5	59.2	61.1	71.5	34.8	37.0	45.5	29.4	45.2	92.0%
GRASP	**44.6**	**35.1**	26.2	**68.4**	41.4	**63.2**	**62.7**	**73.3**	35.1	36.1	43.1	30.7	**46.7**	**94.9%**

Table 2: Comparison between GRASP and structured pruning baselines with post-training compensation under a 25% compression ratio. Results marked with * are reported from (Chen et al., 2024). Bold values indicate the best performance.

lowing the same evaluation protocol as LaCo (Yang et al., 2024).

Main Results. To ensure a fair comparison, we constrain the number of trainable parameters to remain approximately the same across all methods by retaining only 10% of the parameters in each redundant layer and allowing only these parameters to be trainable. Notably, for GRASP's post-training compensation process, we fine-tune the compressed model on Alpaca (Taori et al., 2023) for only one epoch to guarantee efficiency. Additional implementation details are provided in Appendix A.2. As shown in Table 2, GRASP consistently outperforms the best-performing baselines LLM-Streamline on average and achieves a **94.9%** of the original model performance at a 25% compression ratio. In addition to achieving superior accuracy, GRASP also demonstrates more stable and faster convergence during post-training, which we attribute to its preservation of critical singular components.

3.2.3 Comparison with Low-Rank Pruning Methods

Models and Benchmarks. In this section, we further compare our method against state-of-the-art structured low-rank pruning approaches—**FWSVD** (Hsu et al., 2022), **ASVD** (Yuan et al., 2023) and **SVD-LLM** (Wang et al., 2024) on the LLaMA-7B model under various compression ratio. 8 datasets are used as evaluation benchmarks including seven commonsense reasoning datasets as Experiment 1 and one natural language generation (NLG) benchmark **GSM8K** (Cobbe et al., 2021). For all benchmarks, we report the zero-shot accuracy as the evaluation metric.

Main Results. To evaluate the performance and stability of our proposed method, we conduct experiments under various compression ratios ranging from 20% to 50%. Table 3 summarizes the results for different methods. The results demonstrate that our proposed GRASP consistently outperforms the baseline methods on most benchmarks. Specifically, GRASP retains more than 91% of the original performance at a 20% compression ratio and 87% under the compression ratio of 30%. More importantly, with fast and resource-efficient post-training compensation, GRASP enables rapid accuracy recovery, achieving 70% of the original model performance even at a 50% compression ratio.

Furthermore, to assess the generalizability of GRASP across different LLM architectures, we compare its performance against structured low-rank pruning baselines under a 20% compression ratio on four models from two distinct LLM families: LLaMA 2-7B, LLaMA 2-13B, LLaMA 3.1-8B-Instruct, and Mistral-7B. As shown in Figure 3, GRASP consistently outperforms all baseline methods across architectures and exhibits greater robustness across different model families. The only exception is on LLaMA 2-13B, where GRASP

Ratio	Method	Openb.	ARC_e	WinoG.	HeSW	ARC_c	PIQA	MathQA	GSM8K	Average	Percentage
0%	Original	0.28	0.67	0.67	0.56	0.38	0.78	0.27	0.09	0.46	100.0
20%	FWSVD	0.15	0.31	0.50	0.26	0.23	0.56	0.21	0.00	0.28	60.8
	ASVD	**0.25**	0.53	0.64	0.41	0.27	0.68	0.24	0.04	0.38	82.6
	SVD-LLM	0.22	0.58	0.63	0.43	0.29	0.69	0.24	**0.05**	0.39	84.7
	Ours	0.22	0.52	**0.64**	0.43	0.32	0.70	0.24	0.03	0.39	84.7
	Ours*	0.24	**0.59**	0.63	**0.5**	**0.35**	**0.73**	**0.25**	0.04	**0.42**	**91.3**
30%	FWSVD	0.17	0.26	0.49	0.26	0.22	0.51	0.19	0.00	0.26	56.5
	ASVD	0.18	0.43	0.53	0.37	0.25	0.65	0.21	0.00	0.33	71.7
	SVD-LLM	0.20	0.48	0.59	0.37	0.26	0.65	0.22	0.03	0.35	76.1
	Ours	0.19	0.42	0.62	0.39	0.28	0.64	0.23	0.02	0.35	76.1
	Ours*	**0.24**	**0.54**	**0.64**	**0.46**	**0.32**	**0.69**	**0.24**	**0.04**	**0.40**	**87.0**
40%	FWSVD	0.16	0.26	0.51	0.26	0.22	0.53	0.21	0.00	0.27	58.7
	ASVD	0.13	0.28	0.48	0.26	0.22	0.55	0.19	0.00	0.26	56.5
	SVD-LLM	0.19	0.42	0.58	0.33	0.25	0.60	0.21	**0.02**	0.33	71.7
	Ours	0.18	0.37	0.57	0.35	0.27	0.61	0.21	0.01	0.32	69.6
	Ours*	**0.22**	**0.49**	**0.63**	**0.43**	**0.3**	**0.68**	**0.23**	**0.02**	**0.38**	**82.6**
50%	FWSVD	0.12	0.26	0.50	0.26	0.23	0.53	0.20	0.00	0.26	56.5
	ASVD	0.12	0.26	0.51	0.26	0.22	0.52	0.19	0.00	0.26	56.5
	SVD-LLM	0.16	0.33	0.54	0.29	0.23	0.56	0.21	0.00	0.29	63.0
	Ours	0.13	0.29	0.53	0.28	0.23	0.53	0.20	0.01	0.28	60.9
	Ours*	**0.18**	**0.4**	**0.56**	**0.35**	**0.26**	**0.61**	**0.21**	**0.02**	**0.32**	**69.6**

Table 3: Performance of LLaMA-7B compressed by GRASP (Ours* denotes the version with post-training compensation) and low-rank pruning baselines under 20% to 50% compression ratio on seven common sense reasoning datasets (measured by **accuracy**↑) and GSM8K dataset (measured by **Exact Match Accuracy**↑). Percentage represents the proportion of the original model's performance retained by the pruned method. The best performance is marked in bold.

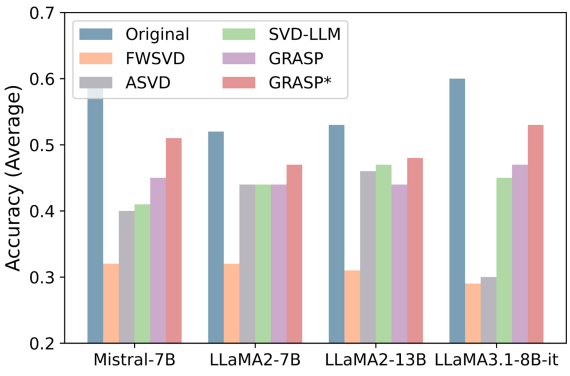

Figure 3: Comparison between our method and low-rank pruning baselines on four different LLMs. Average accuracy is reported across seven commonsense reasoning benchmarks: OpenBookQA, WinoGrande, HellaSwag, ARC-easy, ARC-challenge, PIQA, and MathQA.

slightly underperforms SVD-LLM. However, this performance gap can be quickly recovered through lightweight compensation. The detailed results are provided in Appendix A.6.

3.3 Compression Costs and Inference Efficiency of GRASP

GRASP enables low-cost compression of LLMs while improving inference efficiency on real hardware. To evaluate its acceleration benefits, we measure the throughput (tokens per second) of the original LLaMA2-7B and its GRASP-compressed counterpart under varying sequence lengths and batch sizes. As shown in Figure 5, GRASP consistently improves generation speed and achieves acceleration comparable to direct layer removal. Notably, although GRASP retains a small subset of parameters within redundant layers to mitigate the performance drop caused by layer removal, these retained components are extremely low-rank and incur negligible inference overhead while preserving task performance. Additionally, we measure the compression time of GRASP and other structured pruning baselines when compressing LLaMA2-7B on an NVIDIA A100 GPU under a 25% compression ratio. As reported in Table 4, *Pruning Time* refers to the time required to perform model compression up to the target sparsity, excluding any compensation. *Compensation Time* refers to the time spent on post-compression procedures—such as fine-tuning or parameter updates—aimed at recovering model performance. The results show that GRASP is able to compress the model efficiently while maintaining strong performance, demonstrating its practicality for real-world deployment.

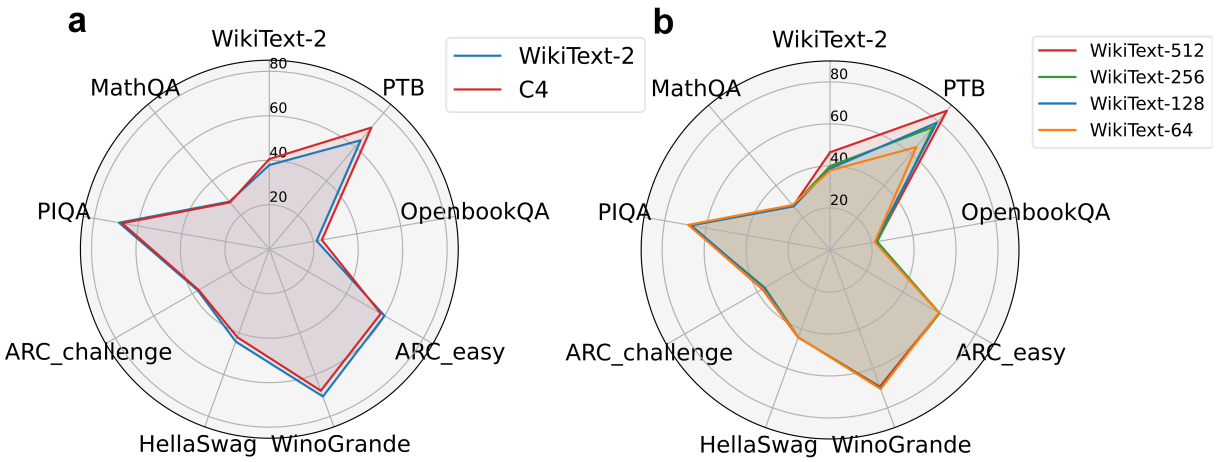

Figure 4: Performance of GRASP on LLaMA3.1-8B-Instruct under 20% compression using (a) different calibration datasets (WikiText-2, C4) and (b) varying amounts of calibration data from WikiText-2. GRASP demonstrates limited sensitivity to calibration data changes, with final task performance varying within 4%.

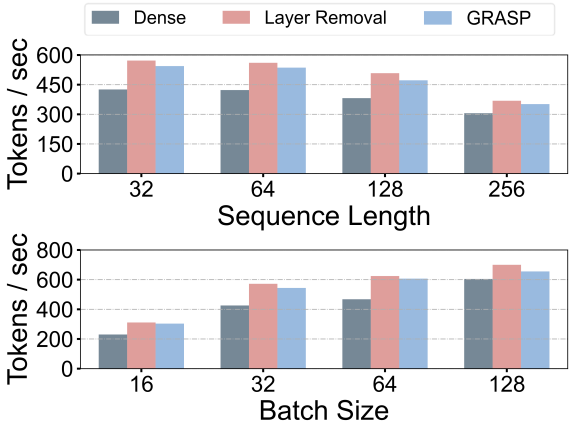

Figure 5: Throughput of LLaMA2-7B and GRASP compressed model under 25% compression ratio on a single A100 GPU. **Top:** Throughput across different sequence lengths (batch size = 32). **Bottom:** Throughput across different batch sizes (sequence length = 32).

Model	Pruning Time (h)	Compensation Time (h)
LaCO	0.05	1.2
SliceGPT	0.6	0.76
LLM-Streamline	0.03	0.7
GRASP	0.16	×
GRASP*	0.16	0.66

Table 4: Compression time of GRASP and structured pruning baselines on LLaMA2-7B under a 25% compression ratio on a single A100 GPU. "×" indicates that GRASP does not require post-training compensation.

3.4 Ablation Study

Calibration Data. We conduct ablation studies to assess the sensitivity of GRASP to the choice and size of the calibration dataset used for gradient-based singular value attribution. Table 5 and Figure 4 present the results on LLaMA3.1-8B-Instruct when varying the calibration dataset and the number of samples drawn from WikiText-2. The results indicate that GRASP remains robust across different calibration dataset choices and performs reliably even with limited calibration data. Remarkably, the method achieves strong performance with as few as 64 samples, demonstrating its effectiveness in low-data regimes.

Ablation Type	Calibration Dataset	WikiText-2	PTB	Average Accuracy
Varying Dataset	WikiText-2	37.86	63.97	47.12
	C4	40.54	71.42	46.17
Varying Number	64	46.5	86.51	47.06
	128	39.91	76.41	46.93
	256	38.73	79.13	46.67
	512	37.86	63.97	47.12

Table 5: Comparison of GRASP using different types and amounts of data for compressing LLaMA3.1-8B-Instruct at a 20% compression ratio. Results are reported on the WikiText-2, PTB datasets (measured by **perplexity**↓) and the average accuracy across seven commonsense reasoning benchmarks: OpenBookQA, WinoGrande, HellaSwag, ARC-easy, ARC-challenge, PIQA, and MathQA.

Retain Ratio r%. We further explore the impact of the retain ratio r on GRASP's performance. As shown in Table 6 and Figure 6, performance improves rapidly when increasing r from 0% to 10%, after which the gains largely saturate. This indicates that a retain ratio of 10% serves as a practical inflection point, providing a strong trade-off between compression and accuracy. In prac-

tice, we therefore adopt a fixed retain ratio of 10%, which consistently yields stable performance across benchmarks while maintaining high compression efficiency. These results further confirm that the selected redundant layers contain substantial redundancy, as even a small fraction of retained singular components suffices to recover most of the original performance.

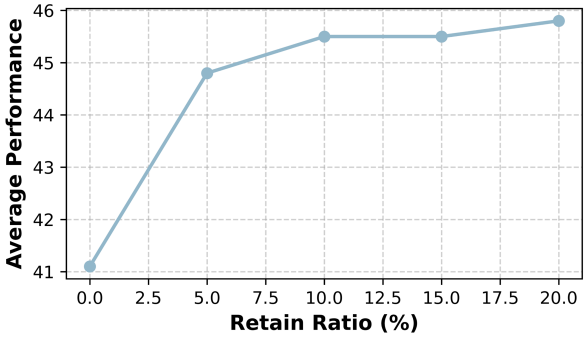

Figure 6: Effect of the retain ratio on average performance when compressing 8 redundant layers of LLaMA2-7B. The results show a sharp improvement up to a 10% retain ratio, after which performance gains plateau, indicating 10% as a practical inflection point.

One-shot Pruning vs Iterative Pruning As detailed in Section 2.3, GRASP processes the redundant layers in a sequential manner, starting from the final redundant layer and proceeding backward through the network. This can be done using one-shot pruning, which identifies and decomposes all redundant layers in a single step, or iterative pruning, which processes layers one at a time to account for interactions between layers. We present the ablation results in Table 7, which shows that both approaches achieve similar performance, with one-shot pruning being more efficient.

Pruning Strategy	WikiText-2	PTB	Average Accuracy	Compression Time(h)
One-shot Pruning	37.86	63.97	47.12	0.16
Iterative Pruning	38.39	72.18	47.13	0.22

Table 7: Comparison of one-shot pruning and iterative pruning for LLaMA3.1-8B-Instruct under 20% compression ratio.

4 Discussion

The effectiveness of GRASP stems from two key design principles that distinguish it from conventional structured pruning methods.

Leveraging Low-Rank Redundancy. GRASP builds on the observation that the redundant layers in LLMs exhibit inherent low-rank characteristics. Rather than removing these layers entirely, which can lead to information loss and degraded performance, GRASP retains a small subset of critical singular directions identified through a gradient-based attribution mechanism. This approach allows the model to retain essential functional capacity while discarding non-informative parameters, leading to efficient yet accurate compression.

Preserving Informative Subspaces for Fast Recovery. By maintaining only the most influential parameters within redundant layers, GRASP avoids the need for costly retraining while facilitating faster convergence when post-training compensation is applied. This contrasts with approaches like LLM-Streamline (Chen et al., 2024) that insert randomly initialized or dense lightweight modules, which require more data and training time to approximate the original function. GRASP's retention of critical low-rank components allows it to preserve the spectral alignment of the original model (Oymak et al., 2019, Kamalakara et al., 2022, Shuttleworth et al., 2024), resulting in more stable optimization dynamics and enhanced sample efficiency during post-training compensation.

5 Related Work

Structured pruning aims to reduce the size and computational cost of large language models (LLMs) by removing entire components such as layers, neurons, or dimensions, while maintaining model performance. Among this, **layer pruning** is a kind of structured pruning technique that eliminates redundant layers within Large Language Models (LLMs). Methods such as ShortGPT (Men et al., 2024) introduce a metric called Block Influence to assess the significance of individual layers, enabling efficient one-shot removal of less important layers. SLEB (Song et al., 2024) improves this by employing an iterative pruning strategy, evaluating the importance of each layer based on the current state of the layer-removed LLMs. LaCo (Yang et al., 2024), on the other hand, proposes a gradual compression approach, progressively merging redundant layers from deeper to shallower parts of the network.

Although effective in reducing model size, layer pruning disrupts representation coherence, leading to performance degradation and increased perplexity, as analyzed by (Liang et al., 2024). To mitigate this, **post-training compensation** methods have been proposed. Kim et al., 2024 introduced Shortened LLaMA, which employs LoRA (Hu et al.,

Retain Ratio	Overall Compression Ratio	Openb.	ARC_e	ARC_c	WinoGrande	PIQA	MathQA	Average
0%	24.0%	20.2	45.1	30.9	61.3	67.0	22.2	41.1
5%	22.8%	22.4	54.9	31.8	66.0	70.0	23.8	44.8
10%	21.6%	23.2	56.1	32.7	66.2	70.7	23.9	45.5
15%	20.4%	23.4	56.3	32.9	65.6	70.7	24.1	45.5
20%	19.2%	23.4	56.5	33.4	65.6	71.5	24.3	45.8

Table 6: Ablation results on LLaMA2-7B by replacing 8 redundant layers with sensitivity-aware singular parameters while varying the retain ratio. The table reports the resulting overall compression ratio and downstream task performance across multiple benchmarks.

2022) to restore pruned models' capabilities. However, LoRA modifies the singular value spectrum, potentially weakening pre-trained features (Shuttleworth et al., 2024). LLM-Streamline (Chen et al., 2024) addresses this by training a lightweight module, such as an FFN or transformer layer, to approximate the removed layers. While effective, these methods impose high computational and data costs, limiting feasibility in resource-constrained settings.

Another line of structured pruning research focuses on low-rank approximation, where Singular Value Decomposition (SVD) is widely used to decompose weight matrices into low-rank structures, typically selecting top-k singular values based on Frobenius norm reconstruction loss. Recent methods have enhanced SVD to reduce LLM compression error. FWSVD (Hsu et al., 2022) incorporates Fisher information to reweight the importance of parameters before applying SVD. ASVD (Yuan et al., 2023) uses activation patterns from a calibration dataset to scale weight matrices, reducing compression-induced activation errors. SVD-LLM (Wang et al., 2024) applies truncation-aware data whitening and layer-wise updates to ensure a direct relationship between singular values and compression loss. Additionally, (Yu and Wu, 2023; Chavan et al., 2024; Ji et al., 2024) present another paradigm for low-rank compression of LLMs, where eigenvalue decomposition is applied to output activations, approximating the output activations with low-rank matrices. However, SVD-based methods require truncating at least 50% of singular values to reduce parameters of square matrices (which are common in LLMs like Llama), which often leads to significant information loss. In contrast to these approaches, GRASP integrates gradient-based attribution into the low-rank decomposition process and replaces redundant layers with only a small fraction of parameters (typically 10%), thereby enabling efficient compression while retaining critical information.

6 Conclusion

In this work, we proposed **GRASP**, a novel compression framework that replaces redundant layers in large language models with a small set of adaptively selected singular parameters. By leveraging the low-rank structure of redundant layers and incorporating gradient-based attribution, GRASP identifies critical components that preserve model functionality with minimal parameter overhead. It operates in a training-free manner and enables efficient post-training recovery with limited data. Extensive experiments across diverse LLM architectures and benchmarks demonstrate that GRASP consistently outperforms existing structured pruning methods in both accuracy and efficiency.

7 Limitations

While GRASP achieves competitive performance and compression efficiency, it also has several limitations that merit further exploration.

First, our method relies on the assumption that layer redundancy can be identified via output similarity (e.g., cosine similarity between hidden states). While effective in practice, this heuristic may overlook more nuanced forms of redundancy that arise from distributed or task-specific behaviors within deeper model layers.

Second, GRASP depends on access to gradient information and a small calibration dataset to compute attribution scores. Although the data requirement is minimal and the method is training-free in its core form, this may limit applicability in strictly black-box or privacy-sensitive settings where gradients or internal representations are inaccessible.

We also note that our experiments are conducted on models up to 13B parameters and primarily in English-language tasks. Extending GRASP to multilingual or much larger-scale models is a promising direction for future work, especially as the scale and diversity of LLMs continue to grow.

Ethical Considerations

Our research adheres to the ACL Code of Ethics, ensuring transparency, responsible use of data, and consideration of potential social impacts. All datasets used in this work are publicly available and have been appropriately cited, ensuring compliance with data usage agreements and privacy regulations.

While GRASP is designed to optimize the efficiency and scalability of large language models, we recognize that such technologies could be misused in applications that may perpetuate harmful biases or deploy models in contexts lacking adequate oversight. To mitigate these risks, we advocate for responsible deployment practices, including thorough testing and monitoring for unintended biases.

Moreover, we acknowledge the computational resources required for training and testing large language models. To minimize environmental impact, we conducted experiments on energy-efficient hardware (NVIDIA A100 GPUs) and report our computational cost transparently. Further details can be found in the Appendix.

References

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Saleh Ashkboos, Maximilian L Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. 2024. Slicegpt: Compress large language models by deleting rows and columns. *arXiv preprint arXiv:2401.15024*.

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

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Arnav Chavan, Nahush Lele, and Deepak Gupta. 2024. Surgical feature-space decomposition of LLMs: Why, when and how? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2389–2400, Bangkok, Thailand. Association for Computational Linguistics.

Xiaodong Chen, Yuxuan Hu, and Jing Zhang. 2024. Compressing large language models by streamlining the unimportant layer. *arXiv preprint arXiv:2403.19135*.

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

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

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass.

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

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International conference on machine learning*, pages 10323–10337. PMLR.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2022. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024. A framework for few-shot language model evaluation.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2023. Knowledge distillation of large language models. *arXiv preprint arXiv:2306.08543*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Yen-Chang Hsu, Ting Hua, Sungen Chang, Qian Lou, Yilin Shen, and Hongxia Jin. 2022. Language model compression with weighted low-rank factorization. In *International Conference on Learning Representations*.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Ting Hua, Xiao Li, Shangqian Gao, Yen-Chang Hsu, Yilin Shen, and Hongxia Jin. 2025. Dynamic low-rank estimation for transformer-based language models. US Patent App. 18/669,413.

Yixin Ji, Yang Xiang, Juntao Li, Qingrong Xia, Zi Ye, Xinyu Duan, Zhefeng Wang, Kehai Chen, and Min Zhang. 2024. Adaptive feature-based low-rank compression of large language models via bayesian optimization. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 4152–4168.

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. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Siddhartha Rao Kamalakara, Acyr Locatelli, Bharat Venkitesh, Jimmy Ba, Yarin Gal, and Aidan N Gomez. 2022. Exploring low rank training of deep neural networks. *arXiv preprint arXiv:2209.13569*.

Bo-Kyeong Kim, Geonmin Kim, Tae-Ho Kim, Thibault Castells, Shinkook Choi, Junho Shin, and Hyoung-Kyu Song. 2024. Shortened LLaMA: A simple depth pruning for large language models. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations. *arXiv preprint arXiv:1704.04683*.

Yann LeCun, John Denker, and Sara Solla. 1989. Optimal brain damage. *Advances in neural information processing systems*, 2.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. *KR*, 2012:13th.

Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. 2023. Cmmlu: Measuring massive multitask language understanding in chinese. *arXiv preprint arXiv:2306.09212*.

Xun Liang, Shichao Song, Zifan Zheng, Hanyu Wang, Qingchen Yu, Xunkai Li, Rong-Hua Li, Yi Wang, Zhonghao Wang, Feiyu Xiong, et al. 2024. Internal consistency and self-feedback in large language models: A survey. *arXiv preprint arXiv:2407.14507*.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. LLM-pruner: On the structural pruning of large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. 2024. Shortgpt: Layers in large language models are more redundant than you expect. *arXiv preprint arXiv:2403.03853*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *International Conference on Learning Representations*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, Brussels, Belgium. Association for Computational Linguistics.

Samet Oymak, Zalan Fabian, Mingchen Li, and Mahdi Soltanolkotabi. 2019. Generalization guarantees for neural networks via harnessing the low-rank structure of the jacobian. *arXiv preprint arXiv:1906.05392*.

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

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8732–8740.

Reece Shuttleworth, Jacob Andreas, Antonio Torralba, and Pratyusha Sharma. 2024. Lora vs full fine-tuning: An illusion of equivalence. *arXiv preprint arXiv:2410.21228*.

Jiwon Song, Kyungseok Oh, Taesu Kim, Hyungjun Kim, Yulhwa Kim, and Jae-Joon Kim. 2024. Sleb: Streamlining llms through redundancy verification and elimination of transformer blocks. *arXiv preprint arXiv:2402.09025*.

Kai Sun, Dian Yu, Dong Yu, and Claire Cardie. 2020. Investigating prior knowledge for challenging chinese machine reading comprehension. *Transactions of the Association for Computational Linguistics*, 8:141–155.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2023. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2024. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Stanford alpaca: An instruction-following llama model.

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

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

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. 2024. Svd-llm: Truncation-aware singular value decomposition for large language model compression. *arXiv preprint arXiv:2403.07378*.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, et al. 2020. Clue: A chinese language understanding evaluation benchmark. *arXiv preprint arXiv:2004.05986*.

Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. 2024. A survey on knowledge distillation of large language models. *arXiv preprint arXiv:2402.13116*.

Yifei Yang, Zouying Cao, and Hai Zhao. 2024. Laco: Large language model pruning via layer collapse. *arXiv preprint arXiv:2402.11187*.

Hao Yu and Jianxin Wu. 2023. Compressing transformers: features are low-rank, but weights are not! In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 11007–11015.

Zhihang Yuan, Yuzhang Shang, Yue Song, Qiang Wu, Yan Yan, and Guangyu Sun. 2023. Asvd: Activation-aware singular value decomposition for compressing large language models. *arXiv preprint arXiv:2312.05821*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Chujie Zheng, Minlie Huang, and Aixin Sun. 2019. Chid: A large-scale chinese idiom dataset for cloze test. *arXiv preprint arXiv:1906.01265*.

Zixuan Zhou, Xuefei Ning, Ke Hong, Tianyu Fu, Jiaming Xu, Shiyao Li, Yuming Lou, Luning Wang, Zhihang Yuan, Xiuhong Li, et al. 2024. A survey on efficient inference for large language models. *arXiv preprint arXiv:2404.14294*.

A Appendix

A.1 The gradient of singular values

For a weight matrix $W \in \mathbb{R}^{m \times n}$ in the selected redundant layers, its differential form can be expressed as:

$$\partial W = \partial U \Sigma V^T + U \partial \Sigma V^T + U \Sigma \partial V^T$$

$$U^T \partial W V = U^T \partial U \Sigma + \partial \Sigma + \Sigma V^T \partial V$$

Since both U and V are orthogonal matrices, we have:

$$U^T U = I_m, \quad V^T V = I_n$$

$$\partial U^T U + U^T \partial U = O_m, \quad \partial V^T V + V^T \partial V = O_n$$

This implies that $U^T dU$ and $dV^T V$ are asymmetric matrices. Therefore, the diagonal elements of $U^T dU \Sigma$ and $\Sigma V^T dV$ are zero, leading to the diagonal elements of $U^T \partial W V$ being:

$$I_k \odot U^T \partial W V = \partial \Sigma$$

where I_k represents the $k \times k$ identity matrix, $\odot$ denotes element-wise multiplication.

For a singular value σ_i, its differential form can be written as:

$$\partial \sigma_i = u_i^T \partial W v_i$$

Since σ_i is a scalar, we have:

$$\begin{aligned} \partial \sigma_i &= \text{tr}(\partial \sigma_i) \\ &= \text{tr}(u_i^T \partial W v_i) \\ &= \text{tr}[(u_i v_i^T)^T \partial W] \end{aligned}$$

thereby, the derivative of σ_i with respect to W is:

$$\frac{\partial \sigma_i}{\partial W} = u_i v_i^T$$

For a calibration dataset D, the gradient of a singular value σ_i with respect to the task loss can be interpreted as the projection of the weight gradient matrix G onto the corresponding singular direction, given by:

$$\frac{\partial L}{\partial \sigma_i} = u_i^T \frac{\partial L}{\partial W} v_i$$

Then, for all the singular values Σ, we have:

$$\frac{\partial L}{\partial \Sigma} = I_k \odot U^T \left(\frac{\partial L}{\partial W} \right) V$$

A.2 Experimental Setup and Hyperparameters Configuration

To ensure a fair comparison, all experimental setups are consistent across all methods. In the following, we describe the experimental setup and hyperparameters configuration in detail.

Hyperparameters Configurations For post-training compensation, all models compressed by GRASP are trained on the Alpaca (Taori et al., 2023) dataset for 1 epoch with a batch size of 32. We use AdamW (Loshchilov and Hutter, 2019) as our optimizer and set the learning rate to 3×10^{-4}. All our experiments are conducted on a single A100 GPU with mixed precision enabled. Table 8 provides the detailed configurations of post-training compensation.

HyperParameters	Setting
Dataset	Alpaca
Huggingface Dataset Path	yahma/alpaca-cleaned
Batch Size	32
Micro Batch Size	4
Epochs	1
Learning Rate	3.00E-04
Max Length	256
Train on Inputs	TRUE
Add EOS Token	FALSE
LoRA-Rank	256
LoRA-Alpha	16
LoRA-Dropout	0.05
LoRA-Target-Modules	q_proj, k_proj, v_proj, o_proj, up_proj, down_proj, gate_proj
Prompt-Template	Alpaca Template

Table 8: Experimental setup and hyperparameters configurations.

A.3 Relationship between Singular Group Retail Ratio and Target Overall Compression Ratio

In GRASP, the retain ratio $r\%$ denotes the proportion of singular components preserved within each weight matrix of the selected redundant layers. The target overall model compression ratio, which is jointly determined by (i) the number of layers selected in Step 1, and (ii) the retain ratio $r\%$ applied in Step 2.

In practice, we adopt a fixed retain ratio—typically 10%—as ablation studies (Section 3.4) show it achieves a favorable trade-off between compression and performance. Table 6 further illustrates how varying the retain ratio influences the overall compression ratio and downstream performance across several benchmarks

Method	Metric	Calibration Data	Need Post-Training	Training Data	Training Dataset Size	Training Module
ShortGPT	Cosine Similarity	WikiText-2	No	None	None	None
LaCo	Cosine Similarity	WikiText-2	Optional	Unpublished	1B	Full Parameters
SliceGPT	PCA	WikiText-2 Alpaca	Optional	Alpaca	5k	Full Parameters
Shortened LLaMA	Taylor Perplexity	BookCorpus	Optional	SlimPajama Alpaca	627B 50k	Full Parameters LoRA-Adapter
LLM-Streamline	Cosine Similarity	WikiText-2	Yes	SlimPajama	30k	Lightweight Network
GRASP	Cosine Similarity	WikiText-2	Optional	Alpaca	50k	Low-rank Modules

Table 9: Comparison of pruning-based LLM compression methods, where the metric indicates the criterion used to identify redundant modules. "Optional" refers to methods that can either work without post-training or recover performance through post-training. Shortened LLaMA consists of two training stages: initial continual pre-training on the SlimPajama dataset, followed by LoRA fine-tuning on the Alpaca dataset.

when compressing 8 layers of LLaMA2-7B. These results highlight the high redundancy of the selected layers: even a small fraction of retained singular components is sufficient to recover most of the original performance.

A.4 Comparison of Concurrent Structured Pruning Methods

We provide a detailed comparison of concurrent structured pruning LLM compression methods, and the results are summarized in Table 9.

A.5 Detailed Results on Commonsense Reasoning Benchmarks

In this section, we provide detailed results for GRASP and baseline methods on the commonsense reasoning benchmarks using LLaMA2-7B, LLaMA2-13B, and Mistral-7B. These results extend the main experiments presented in Table 1, offering a comprehensive view of GRASP's performance across different model architectures.

Table 10 reports accuracy without post-training compensation. The results demonstrate that GRASP consistently maintains strong accuracy and demonstrates robustness across model scales and families.

A.6 More Results on Other Models

To further validate the generalizability of GRASP across diverse LLM architectures, we present additional results under a 20% compression ratio on three representative models beyond LLaMA-7B, LLaMA2-13B, LLaMA3.1-8B-Instruct, and Mistral-7B. Table 11 summarizes the performance comparison between GRASP and structured low-rank pruning baselines, including FWSVD, ASVD, and SVD-LLM, across eight evaluation benchmarks.

Consistent with our findings in the main text, GRASP achieves superior or comparable accuracy across most benchmarks. While SVD-LLM slightly outperforms GRASP on LLaMA2-13B, our method demonstrates stronger robustness across model families and benefits from more stable post-compensation recovery. These results highlight GRASP's effectiveness as a generalizable and architecture-agnostic compression strategy.

A.7 Evaluation Results on LongBench

In this section, we present the detailed results of LLaMA3.1-8B-Instruct and its compressed version by GRASP under 20% compression ratio on LongBench, which are presented in Table 12. The results illustrate that GRASP with post-training compensation still maintains superior performance on long-form reasoning and complex generative tasks.

A.8 Robustness of GRASP towards different calibration Dataset

In this section, we provide details of the ablation studies conducted to investigate the impact of calibration datasets and the amount of data used for singular value gradient attribution. Specifically, we selected 512 samples from WikiText-2 (Merity et al., 2017) and C4 (Raffel et al., 2020) as calibration data to assess the performance of GRASP when compressing LLaMA3.1-8B-Instruct under 20% compression ratio. Additionally, we selected 64, 128, 256 and 512 samples from WikiText-2 to examine the robustness of GRASP to the change in the number of calibration data. All calibration data were randomly selected from the training splits of the downstream datasets, ensuring no data leakage.

Model	Method	Ratio	Openb.	ARC_e	WinoG.	HeSW	ARC_c	PIQA	MathQA	Average
Mistral-7B-v0.1	Dense	0.0%	0.33	0.81	0.74	0.61	0.50	0.81	0.36	0.59
	LaCo	21.1%	0.20	0.35	0.58	0.26	0.25	0.53	0.24	0.34
	ShortGPT	21.1%	0.19	**0.57**	**0.68**	**0.46**	0.37	**0.71**	**0.26**	**0.46**
	SliceGPT	20.0%	0.19	0.51	0.59	0.35	0.25	0.61	0.23	0.39
	GRASP	20.0%	**0.21**	0.56	**0.68**	0.43	**0.38**	0.67	**0.26**	**0.46**
LLaMA2-7B	Dense	0.0%	0.32	0.69	0.67	0.57	0.40	0.78	0.28	0.53
	LaCo	18.1%	**0.26**	0.48	0.59	0.42	0.32	0.69	**0.24**	0.43
	ShortGPT	21.1%	0.23	0.49	**0.63**	0.42	0.31	0.68	0.23	0.43
	SliceGPT	21.5%	0.22	**0.54**	0.61	0.37	0.28	0.63	0.23	0.41
	GRASP	21.6%	0.24	**0.54**	**0.63**	**0.43**	**0.33**	**0.71**	0.23	**0.44**
LLaMA-2-13B	Dense	0.0%	0.32	0.73	0.70	0.60	0.46	0.79	0.30	0.56
	LaCo	19.5%	0.28	0.52	0.63	0.43	0.33	0.70	**0.25**	0.45
	ShortGPT	22.1%	0.24	0.50	0.63	0.46	0.33	0.7	0.24	0.44
	SliceGPT	20.0%	**0.29**	0.59	0.65	0.39	0.32	0.64	0.24	0.45
	GRASP	20.0%	0.26	**0.61**	**0.66**	**0.47**	**0.35**	**0.73**	0.24	**0.47**
LLaMA3.1-8B-Instruct	Dense	0.0%	0.34	0.82	0.74	0.59	0.52	0.80	0.39	0.60
	LaCo	19.0%	**0.26**	0.49	0.65	0.33	0.30	0.65	**0.30**	0.42
	ShortGPT	21.7%	0.21	0.57	0.66	0.42	0.32	0.67	0.26	0.44
	SliceGPT	20.0%	0.15	0.43	0.51	0.30	0.23	0.58	0.22	0.35
	GRASP	20.0%	0.22	**0.60**	**0.70**	**0.44**	**0.37**	**0.69**	0.28	**0.47**

Table 10: Zero-shot performance of GRASP and pruning-based without post-training baselines under 20% compression ratio. Results are reported on seven reasoning datasets (individual and average accuracy). Bold values indicate the best performance.

Method	**Mistral-7B**		**LLaMA2-7B**		**LLaMA2-13B**		**LLaMA3.1-8B-Instruct**	
	PPL ↓ (WikiText-2)	Acc ↑ Average	PPL ↓ (WikiText-2)	Acc ↑ Average	PPL ↓ (WikiText-2)	Acc ↑ Average	PPL ↓ (WikiText-2)	Acc ↑ Average
Original	5.25	0.59	5.68	0.52	5.47	0.53	7.21	0.60
FWSVD	6357	0.32	1727	0.32	2360	0.31	3256.7	0.29
ASVD	19.28	0.4	11.14	0.44	9.70	0.46	2443.99	0.30
SVD-LLM†	**10.21**	0.41	**7.94**	0.44	**8.50**	0.47	-	-
Ours	18.42	0.45	14.79	0.44	16.12	0.44	37.86	0.47
Ours*	11.62	**0.51**	10.19	**0.47**	9.59	**0.48**	**14.13**	**0.53**

Table 11: Perplexity(↓) of GRASP and low-rank pruning baselines on the WikiText-2 datasets and the average accuracy(↑) on seven common sense reasoning datasets for four different LLMs under 20% compression ratio. "†" indicates that we refer to the results in the original paper..The best performance is marked in bold.

As shown in Figure 4, we can observe that GRASP consistently achieves strong performance, indicating that our method is robust to variations in both the calibration dataset and the number of data. Tables 13 and 14 summarize the results of GRASP when compressing LLaMA3.1-8B-Instruct with different calibration datasets (WikiText-2, C4) and varying numbers of calibration data.

A.9 Comparison with Unstructured Pruning Methods

To provide a more comprehensive picture of model compression, we have included comparisons with two widely used unstructured pruning methods, SparseGPT (Frantar and Alistarh, 2023) and Wanda (Sun et al., 2023) in our experiments. While unstructured pruning introduces fine-grained sparsity in weight matrices, it typically requires dedicated sparse formats to realize storage or runtime benefits, which complicates direct comparison under the same compression ratio. To ensure a fair evaluation, we adopt the commonly used 2:4 sparsity pattern (i.e., 50% sparsity) for both methods, and compare them against GRASP at a 20% structured compression ratio. The results are summarized in Table 15, indicating that GRASP achieves competitive or superior performance against unstructured pruning methods across all benchmarks.

Model	Summarization				Few-shot Learning				Synthetic Task		
	1-1	**1-2**	**1-3**	**1-4**	**1-1**	**2-1**	**2-2**	**2-3**	**2-4**	**3-1**	**3-2**
LLaMA3.1-8B-Instruct	28.35	20.04	25.85	14.51	63	51.3	39.29	16.5	1.64	9.67	17.17
GRASP	**25.76**	**19.41**	25.97	**8.99**	**59.5**	67.44	**38.41**	**18**	1	**10**	**18**

Model	One-Doc QA				Multi-Doc QA				Code Completion		Average
	4-1	**4-2**	**4-3**	**4-4**	**5-1**	**5-2**	**5-3**	**5-4**	**6-1**	**6-2**	**ALL**
LLaMA3.1-8B-Instruct	9.96	6.04	18.75	14.27	11.27	11.57	7.65	34.16	56.47	51.34	26.09
GRASP	15.32	**20.47**	34.52	16.46	31.33	**26.18**	**10.28**	**29.86**	**33.74**	**38.09**	26.26

Table 12: Performance comparison of LLaMA3.1-8B-Instruct and its compressed version by GRASP under 20% compression ratio on LongBench. The datasets are grouped as follows: **(1-1 to 1-4)** denote GovReport, QMSum, MultiNews, and VCSUM; **(2-1 to 2-4)** denote TREC, TriviaQA, SAMSum, and LSHT; **(3-1 to 3-3)** denote PassageCount, PassageRetrieval-en, and PassageRetrieval-zh; **(4-1 to 4-4)** denote NarrativeQA, Qasper, MultiFieldQA-en, and MultiFieldQA-zh; **(5-1 to 5-4)** denote HotpotQA, 2WikiMultihopQA, MuSiQue, and DuReader; **(6-1 to 6-2)** denote LCC and RepoBench-P.

Calibration Dataset	WikiText-2	PTB	Openb.	ARC_e	WinoG.	HeSW	ARC_c	PIQA	MathQA	Average
WikiText-2	37.86	63.97	21.6	59.85	70.48	44.21	37.12	68.66	27.94	47.12
C4	40.54	71.42	24	57.91	67.72	42.11	36.69	67.25	27.5	46.17

Table 13: Zero-shot performance of LLaMA3.1-8B-Instruct compressed by GRASP under 20% compression using 512 samples from WikiText-2 and C4 as calibration datasets.

Calibration Dataset	WikiText-2	PTB	Openb.	ARC_e	WinoG.	HeSW	ARC_c	PIQA	MathQA	Average
WikiText-2-64	46.5	86.51	22.6	59.97	69.3	44.36	37.29	68.5	27.37	47.06
WikiText-2-128	39.91	76.41	22.8	60.23	69.93	44.24	35.92	67.9	27.5	46.93
WikiText-2-256	38.73	79.13	21.8	59.89	70.24	44.23	36.26	67.19	27.07	46.67
WikiText-2-512	37.86	63.97	21.6	59.85	70.48	44.21	37.12	68.66	27.94	47.12

Table 14: Zero-shot performance of LLaMA3.1-8B-Instruct compressed by GRASP under 20% compression with varying calibration data sizes (64, 128, 256, 512) from WikiText-2.

Method	Openb.	ARC_e	ARC_c	WinoG.	HeSW	PIQA	MathQA	Average
Dense	32.12	72.69	39.87	67.33	57.16	78.46	26.02	53.38
SparseGPT(2:4)	24.40	64.10	30.80	66.61	43.09	70.73	24.25	46.28
wanda(2:4)	24.60	62.75	30.80	62.67	41.38	70.29	23.62	45.16
GRASP	24.80	56.02	32.68	66.14	42.57	70.74	23.92	45.27
GRASP*	28.20	68.73	37.63	67.88	50.98	72.47	24.19	50.01

Table 15: Comparison between GRASP and unstructured pruning methods under the 2:4 sparsity pattern (50% sparsity). GRASP is applied at a 20% structured compression ratio. GRASP* denotes the variant with light performance compensation