# Sorting, Truncation, and Preservation: A Unified Pipeline for LLM Low-Rank Approximation

Anonymous ACL submission

## Abstract

Low-rank decomposition serves as a promising compression paradigm for large language models, however, rank allocation remains challenging: manual rules lack generalizability, and learning-based approaches incur heavy computational overhead. To address these issues, we formulate global low-rank allocation as a unified sorting-and-truncation pipeline, and score each singular component via dual criteria: **Local** singular energy ratio that quantifies the intrinsic importance within the decomposed parameter matrix and **Global** functional importance (measured by input-output cosine similarity) that evaluates the functional significance of decomposed modules. We verify the strong correlation between high input-output cosine similarity and low effective rank through geometric interpretation and experimental validation. Furthermore, we propose rank-preserving fine-tuning, which performs direct LoRA tuning on decomposed weights and avoids extra information loss caused by re-truncation in conventional merging pipelines. Empirical results confirm that our method delivers sustained performance enhancements when combined with models featuring distinct decomposition schemes, model sizes and architectural designs, e.g. in one-shot compression without further fine-tuning, our method reduces perplexity by up to 50% compared with uniform and heuristic allocation baselines.

## 1 Introduction

Large language models (LLMs) have achieved state-of-the-art results across various domains, including question answering (Lin et al., 2025b; Molfese et al., 2025), code generation (Crupi et al., 2025; Zhang et al., 2025), and mathematical reasoning (Ahn et al., 2024; Setlur et al., 2024). However, their massive parameter scale and overwhelming computational overhead severely hinder real-world deployment. As a mainstream structured compression technique, low-rank approximation significantly accelerates inference while preserving the internal architecture of the model. Recent empirical studies have confirmed the inherent low-rank properties of transformer weights (Smith et al., 2025; Saha et al., 2024), positioning low-rank compression as a highly promising and rapidly advancing direction for efficient LLMs.

Low-rank approximation for model mainly includes: matrix decomposition and rank truncation. Although existing works have achieved substantial progress in decomposition designs (Jaderberg et al., 2014; Hsu et al., 2022; Zhang and Papyan, 2025), the rank truncation problem, which assigns proper rank budgets to individual parameter matrices, still lacks in-depth and systematic exploration. Current rank allocation schemes can be summarized into three types: (1) uniform allocation (Wang et al., 2025) that assigns identical rank to all layers, which ignores parameter importance and leads to sub-optimal performance; (2) heuristic strategies (Li et al., 2024) that allocate more budget to matrices with strong low-rank properties via empirical observation, yet suffer from limited generalization; and (3) end-to-end learning methods (Qinsi et al., 2025) that relax discrete rank selection into continuous mask optimization, which incurs substantial extra training overhead and exhibits strong sensitivity to parameter initialization. Consequently, developing a low-cost, generalizable approach to rank allocation remains a critical open challenge.

Ideally, the optimal rank allocation can be obtained by evaluating model performance over all combinations of reserved singular vectors for all parameter matrices, yet such exhaustive combinatorial search is computationally intractable and infeasible for large-scale language models. Instead, we simplify the intractable global optimization problem into a concise *Sorting-and-Truncation (S&T)* pipeline over all singular vectors derived from full model weight matrices, on the premise of constructing a reliable scalar metric to quantify the impor-

tance of each singular component. Different from previous rank assignment strategies, our paradigm integrates both structural reconstruction and functional impact for comprehensive measurement.

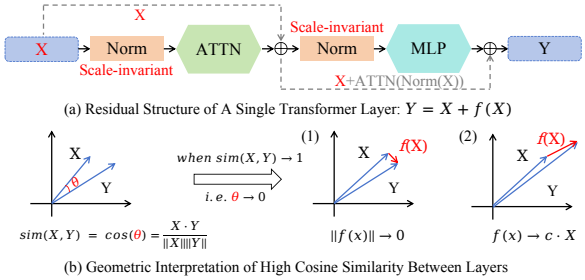


Figure 1: Motivation illustration. (a): the composite residual structure allows input $X$ to directly flow to output $Y$, yielding the layer formulation $Y = X + f(X)$. (b): a high input-output cosine similarity implies $f(X)$ is either near zero or collinear with $X$, both corresponding to low effective rank and high compressibility.

Our core insight is that the overall importance of a singular vector depends on two critical factors. The first is *local structural significance*, which measures the singular vector's contribution to reconstructing the original weight matrix. Following the Eckart-Young-Mirsky theorem (Eckart and Young, 1936), optimal low-rank reconstruction is driven by the magnitude of singular values, so we quantify this using the normalized squared singular value energy ratio, $\sigma_k^2 / \sum_j \sigma_j^2$. The second is *global functional importance*, which assesses the host module's impact on the overall network. To evaluate this without heavy computation, we exploit the intrinsic residual architecture of modern pre-norm LLMs (Touvron et al., 2023a,b; Grattafiori et al., 2024; Yang et al., 2025; DeepSeek-AI, 2026), formulated as $Y = X + f(X)$ (Figure 1(a)). Due to the widespread use of scale-invariant normalization (e.g., RMSNorm (Zhang and Sennrich, 2019)), the semantic representation of information is strictly encoded in the directional vector space rather than its magnitude (Ethayarajh, 2019). Consequently, we use layer-wise input-output cosine similarity computed on a small calibration set as a lightweight proxy to quantify module redundancy. Unlike previous methods that strictly use similarity for layer pruning (Chen et al., 2025; Men et al., 2024; Zhao et al., 2025), we establish it as a reliable indicator of functional effective rank. As illustrated in Figure 1(b), a high input-output similarity implies that the layer transformation is either negligible (singular values $\approx 0$) or linearly correlated (effective rank $\approx 1$) with input features, both corresponds to a low effective rank.

Furthermore, existing low-rank compression methods suffer from a critical flaw when adapting to LoRA fine-tuning (Hu et al., 2022). Since low-rank models are decomposed into two small factor matrices rather than full-rank weights, they fail to directly merge LoRA incremental parameters through standard weight reparameterization. Conventional schemes (Li et al., 2024) first reconstruct full-rank matrices by multiplying factors, integrate LoRA updates, then re-decompose and re-truncate to the target rank, which causes unavoidable information loss and extra computation. To address this, we propose *Rank-Preserving Fine-Tuning (RPFT)*, which splits the allocated rank $k$ into a tunable part $k_{\text{LoRA}}$ and a frozen part $k - k_{\text{LoRA}}$. This design eliminates re-decomposition and perfectly preserves fine-tuned information without truncation loss, enabling efficient adaptation and no extra information loss.

We evaluate our approach by instantiating a full compression and adaptation pipeline, which we name *UniRank*. UniRank is designed as a modular framework: S&T provides a plug-and-play rank allocation strategy that replaces uniform or learning-based assignment, while RPFT enables post-compression tuning without exceeding the predefined rank budget. We integrate UniRank with multiple state-of-the-art decomposition methods and show that it consistently achieves superior or competitive performance compared to existing allocation schemes, with significantly enhanced generalizability across architectures.

- We propose a global rank allocation framework that assigns low-rank budgets via a low-cost sorting-and-truncation pipeline.
- We provide both geometric interpretation and experimental valiadition that connects high input-output cosine similarity to low effective rank, offering a lightweight criterion for low-rank compression.
- We introduce Rank-Preserving Fine-Tuning, a novel fine-tuning paradigm for low-rank models that enables efficient integration of LoRA updates on pre-decomposed factor matrices.
- Extensive experiments on language modeling and downstream tasks demonstrate that our UniRank framework consistently outper-

forms existing rank allocation and compression methods.

## 2 Related Works

Model compression mitigates LLM inference overhead, with sparsification split into unstructured and structured approaches. Unstructured sparsity (Frantar and Alistarh, 2023; Huang et al., 2025) trims individual weights, while structured sparsity (Ashkboos et al., 2024; Men et al., 2024) prunes contiguous units for hardware efficiency. Low-rank compression decomposes original linear layers into two low-rank linear sub-modules, featuring inherent structural properties and favorable hardware compatibility.

Existing low-rank compression methods prioritize elaborate parameter decomposition. Recent findings (Yu and Wu, 2023; Jaiswal et al., 2025) demonstrate that activations, instead of model weights, carry intrinsic low-rank structures, motivating calibration-based activation decomposition as a mainstream research direction. Along this line of research, representative efforts explore diversified optimization paradigms to refine feature decomposition and model linearization, including covariance distribution alignment (Ji et al., 2024; Wang et al., 2025), holistic structural decomposition (Lin et al., 2025a; Saha et al., 2024; Smith et al., 2025) and outlier-aware pruning (Zhang and Papyan, 2025).

Rank allocation is another critical yet underexplored dimension in low-rank compression. Existing rank assignment strategies can be categorized into three mainstream paradigms: 1. Uniform allocation (Yu and Wu, 2023; Wang et al., 2025) uniformly reserves identical rank proportions for all parameters, widely serving as a standard baseline. 2. Heuristic methods conduct coarse-grained rank assignment across layers (Lu et al., 2025; Mi et al., 2025; Tian et al., 2026) and parameter types (Li et al., 2024) based on activation statistics. Though computationally efficient, they cannot accommodate fine singular-level properties. 3. Learning-based approaches (Gao et al., 2024b; Qinsi et al., 2025; Li et al., 2026) support singular-level assessment yet incur substantial overhead. Visualization observations (as shown in Figure.4) further verify their initialization sensitivity and conservative assignment tendency. In this paper, we explore effective singular-level importance measurement, achieving efficient computation and better alignment of feature distributions across diverse modules and parameters

## 3 Method

In this section, we present the proposed **UniRank** framework. We detail the **Sorting-and-Truncation (S&T)** module for adaptive rank budget assignment. Then, we describe the **Rank-Preserving Fine-Tuning (RPFT)** scheme designed to efficiently fine-tune compressed models without extra information loss. Figure 2 illustrates how UniRank works.

### 3.1 Decomposition

Given a weight matrix $\mathbf{W} \in \mathbb{R}^{m\times n}$ from LLM `linear` modules (we omit layer and type indices for clarity), we perform singular value decomposition (SVD) to factorize it into low-rank components:

$$\mathbf{W} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^\top, \tag{1}$$

where $\mathbf{U} \in \mathbb{R}^{m\times r}$ and $\mathbf{V} \in \mathbb{R}^{n\times r}$ are orthogonal matrices of left and right singular vectors, $\mathbf{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_r)$ with singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$, and $r = rank(W) \leq \min(m, n)$ is the full rank. A rank-$k$ approximation ($k < r$) is $\widetilde{\mathbf{W}}_k = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^\top$, with approximation error proportional to the discarded singular values.

**Sorting-and-Truncation (S&T).** We introduce **S&T**, a principled and architecture-agnostic strategy for adaptive rank allocation in low-rank decomposed LLMs. S&T operates by constructing a unified saliency metric across all parameters, enabling globally optimal rank assignment under budget constraint.

In Section 1, we have established that high input-output cosine similarity implies intrinsically low effective rank for a functional module, based on the geometric interpretation. To further validate this intuition empirically, we conduct a layer-wise correlation analysis: we measure the performance drop caused by rank truncation while keeping 90% singular energy per layer, and compare it with $1 - \mathbb{E}[\cos(x, y)]$ computed on the calibration set. As illustrated in Figure 3, the two metrics exhibit a strong linear correlation with a Pearson coefficient of **0.8792**, corroborating that cosine similarity serves as a reliable indicator of layer sensitivity.

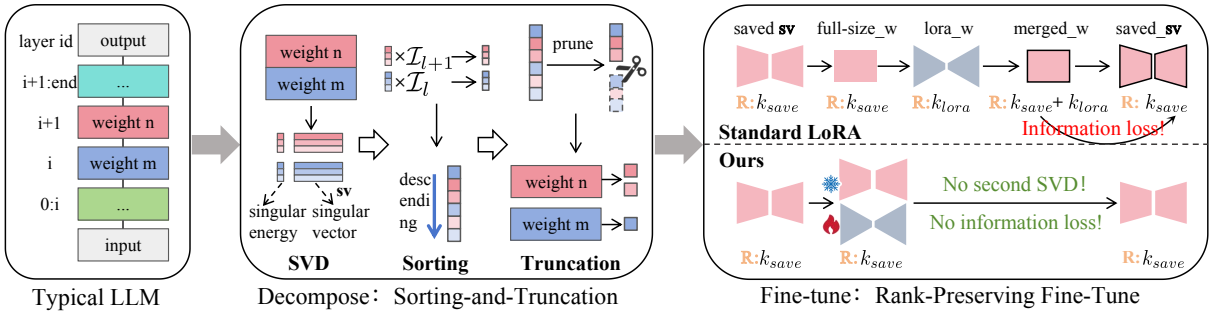


Figure 2: Main framework of UniRank. This figure illustrates the workflow of decomposition and fine-tuning. For the decomposition process, S&T ranks the singular values, then determines the singular vectors to be retained based on target sparsity. For the fine-tuning process, RPFT splits the retained singular vectors into two groups: frozen (denoted by snowflake) and the trainable group (denoted by flame).

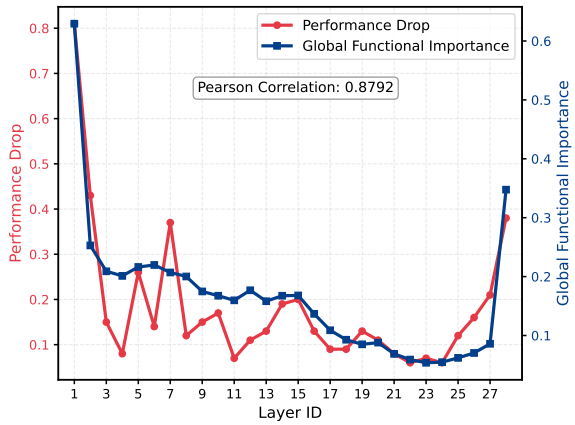


Figure 3: Quantified relationship between performance drop (perplexity) and layer-wise global functional importance. A strong positive linear correlation is confirmed (Pearson $r = 0.8792$).

Accordingly, we define *global functional importance* as

$$\mathcal{I}_l = 1 - \mathbb{E}_{x \sim \mathcal{D}_c}[\cos(f_{l-1}(x), f_l(x))],$$

where $f_l(x)$ denotes the output of layer $l$, and $\cos(\cdot, \cdot)$ denotes average cosine similarity over tokens. Intuitively, layers that induce larger feature transformations are assigned higher importance.

For local structural significance, we adopt the *singular energy ratio*: given decomposition results $U\Sigma V^\top$, each singular component $(u_i, \sigma_i, v_i)$ contributes normalized energy $\sigma_i^2 / \|\Sigma\|_F^2$, reflecting how much variance it preserves.

S&T unifies these two factors into a aggregated importance score:

$$s_{l,i} = \mathcal{I}_l \cdot \frac{\sigma_{l,i}^2}{\|\Sigma_l\|_F^2}, \tag{2}$$

which quantifies the importance of the $i$-th singular triplet in layer $l$. All scores across the network are aggregated and sorted in descending order. Notably, this pipeline requires no gradient-based optimization or handcrafted architectural priors. Furthermore, S&T is fully decoupled from the base decomposition method and can be applied post-hoc to any SVD-based compressed model, making it a plug-and-play module for boosting rank efficiency.

**Accumulative Rank Preservation** Following global sorting of all singular components, we observe that singular vectors from distinct weight matrices introduce heterogeneous parameter overhead. For an $m \times n$ weight matrix, retaining one singular vector pair consumes $m + n$ parameters. To address this variable parameter cost across matrices of different dimensions, we employ an accumulative rank preservation strategy. We first calculate the reserved parameter volume based on the total model parameters and target sparsity ratio, then iteratively select and aggregate the top singular vector pairs in descending order of importance until the total parameter consumption precisely reach the predefined budget.

### 3.2 Fine-Tuning

Low-rank compression decomposes weights into $\widetilde{\mathbf{W}} = \mathbf{A}\mathbf{B}^\top$, with $\mathbf{A} \in \mathbb{R}^{m \times k}$ and $\mathbf{B} \in \mathbb{R}^{n \times k}$.

**When LoRA meets Low-Rank Weights** In standard LoRA, a trainable low-rank incremental matrix $\Delta\mathbf{W} = \mathbf{A}_{\text{lora}}\mathbf{B}_{\text{lora}}^\top$ is added to a frozen pre-trained weight $\mathbf{W}$, and during inference, $\Delta\mathbf{W}$ is merged into $\mathbf{W}$ via re-parameterization, i.e $\widetilde{\mathbf{W}} = \mathbf{W} + \Delta\mathbf{W}$. However, for low-rank compressed model, the original matrix is already decomposed into low-rank matrices $\mathbf{A}\mathbf{B}^\top$, merging $\Delta\mathbf{W}$ would require reconstructing a full matrix $[\mathbf{A}\mathbf{B}]^\top + \Delta\mathbf{W}$, and another SVD on the merged matrix, which

would cause extra information loss (Appendix A.2 gives an detailed analysis on rank dynamics).

**Rank-Preserving Fine-Tuning (RPFT)** We present RPFT to fix this issue, which optimizes partial singular components under constant reserved rank. Each compressed weight satisfies $\widetilde{\mathbf{W}} = \mathbf{A}\mathbf{B}^\top = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^\top$. we divide retained singular vectors into two sets: (1) $\mathcal{I}_{\text{train}}$: a small number (typically 8 or 16) of singular vectors with low importance scores. (2) $\mathcal{I}_{\text{fix}}$: the remaining majority of singular vectors.

We directly optimize trainable vectors. The fine-tuned weight is written as:

$$\widetilde{\mathbf{W}}_{\text{finetuned}} = \sum_{i \in \mathcal{I}_{\text{fix}}} \sigma_i \mathbf{u}_i \mathbf{v}_i^\top + \sum_{j \in \mathcal{I}_{\text{train}}} \sigma_j^{(t)} \mathbf{u}_j^{(t)} (\mathbf{v}_j^{(t)})^\top, \tag{3}$$

where superscript $(t)$ denotes trainable parameters. The rank stays bounded by $k$ to avoid rank growth and information loss. RPFT uses sparse trainable units and discards re-decomposition, effectively reducing computation cost.

# 4 Experiments

## 4.1 Experimental Setup

**Models** We conduct experiments on diverse architectures (MHA: Llama2 (Touvron et al., 2023b) and GQA: Llama3 (Grattafiori et al., 2024)) and scales (3B, 7B, 8B).

**Baseline Methods.** We compare against a diverse set of SOTA compression techniques: 1) *Parameter/Layer Pruning*: ShortGPT (Men et al., 2024), SliceGPT (Ashkboos et al., 2024), and Shortened-Llama (Kim et al., 2024); 2) *Token Pruning*: MoD (Raposo et al., 2024), D-LLM (Jiang et al., 2024), and SkipGPT (Zhao et al., 2025); 3) *Low-rank Pruning*: LoRAP (Li et al., 2024), AFM (Yu and Wu, 2023), Dobi(Qinsi et al., 2025)). Detailed configuration are reported in Appendix A.1

**Evaluation Benchmarks.** *Reasoning*:Accuracy on BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), HellaSwag (Zellers et al., 2019), Winogrande (Sakaguchi et al., 2021), ARC-E/ARC-C (Clark et al., 2018), and OBQA (Mihaylov et al., 2018). *Modeling*: Perplexity (PPL) on WikiText-2 (Merity et al., 2017). All tasks are evaluated by lm-eval 0.4.9 (Gao et al., 2024a).

Table 1: WikiText perplexity under zero-shot compression (lower is better).

| Decom. | Manual | S&T(Ours) | Reduction |
|---|---|---|---|
| SVD | 762.52 | 291.48 | 61.8% |
| AFM | 146.63 | 86.88 | 40.8% |
| AWSVD | 127.73 | 55.83 | 56.3% |

## 4.2 Zero-Shot Direct Compression

We evaluate the zero-shot compression performance of our S&T rank allocation framework against traditional hand-crafted rank assignment strategies. All experiments are conducted without fine-tuning, exclusively reflecting the efficacy of rank allocation in preserving inherent model knowledge. As shown in Table 1, S&T achieves consistent and substantial perplexity reductions across all decomposition paradigms, with an average relative improvement of **52.9%**. Even for weak baseline decomposition methods (e.g., naive weight SVD), S&T effectively alleviates catastrophic performance degradation, lowering the perplexity from 762.52 to 291.48 and demonstrating strong robustness to diverse factorization techniques.

## 4.3 Full Performance Evaluation

As shown in Table 2, our UniRank achieves competitive or superior performance against state-of-the-art baselines based on the Llama3.1-8b model. It obtains an average reasoning accuracy of 66.0%, surpassing LoRAP and ShortGPT by 3.0% and 4.0%, respectively. On WikiText, UniRank yields a perplexity of 10.22, ranking second among eight baseline methods and only slightly lower than SliceGPT (9.22%). Experimental results reveal that parameter-sparsity methods substantially outperform token-sparsity approaches, including MoD, D-LLM and SkipGPT, under moderate 25% compression, indicating that parameter redundancy dominates token redundancy at low sparsity. The performance gap between the two paradigms narrows at a higher 40% sparsity rate. Even so, UniRank remains competitive, achieving a reasoning accuracy of 59.13% and a perplexity of 14.03, which is comparable to the best baseline results.

Furthermore, UniRank consistently outperforms the LoRAP decomposition method at both 25% and 40% sparsity. At 25% sparsity, our method improves the average accuracy of seven reasoning tasks by 3% and reduces WikiText perplexity by

Table 2: Performance comparison of different pruning methods on reasoning and language modeling tasks at sparsity levels of 25% and 40%. For reasoning tasks, we report accuracy (%); higher is better. For Wikitext-2 (WT2), we report perplexity (PPL); lower is better. The best results under each sparsity level are highlighted in **bold** and the second best are underlined.

(a) Sparsity = 25%

| Method | Reasoning (Acc. ↑) | | | | | | | | WT2 (PPL ↓) |
|---|---|---|---|---|---|---|---|---|---|
| | BoolQ | OBQA | PIQA | WinoG. | Hella. | ARC-C | ARC-E | AVG | |
| Dense | 82.14 | 44.60 | 81.07 | 77.43 | 81.89 | 57.68 | 84.81 | 72.80 | 7.33 |
| SliceGPT | 72.39 | 34.40 | 66.70 | 61.56 | 56.96 | 31.48 | 50.08 | 53.37 | **9.22** |
| Shortened-llama | 71.19 | 37.40 | 73.72 | **71.82** | 69.56 | 44.45 | 66.88 | 62.15 | 10.32 |
| ShortGPT | 72.05 | 38.40 | 73.94 | 70.96 | 69.23 | 43.86 | 68.01 | 62.35 | 11.13 |
| MoD | 50.28 | 31.60 | 64.25 | 52.41 | 50.44 | 28.24 | 37.67 | 44.98 | 34.21 |
| D-LLM | 50.36 | 30.20 | 57.40 | 52.49 | 37.64 | 28.16 | 37.12 | 41.91 | 40.12 |
| SkipGPT | 70.67 | 29.60 | 56.96 | 62.83 | **74.22** | **49.91** | **78.79** | 60.43 | 10.53 |
| LoRAP | 74.50 | 37.00 | 76.06 | 65.54 | 70.33 | 44.97 | 73.23 | 63.09 | 12.82 |
| + UniRank | **76.87** | **40.40** | **78.31** | 69.90 | 73.77 | 47.67 | 75.49 | **66.05** | 10.29 |

(b) Sparsity = 40%

| Method | Reasoning (Acc. ↑) | | | | | | | | WT2 (PPL ↓) |
|---|---|---|---|---|---|---|---|---|---|
| | BoolQ | OBQA | PIQA | WinoG. | Hella. | ARC-C | ARC-E | AVG | |
| Dense | 82.14 | 44.60 | 81.07 | 77.43 | 81.89 | 57.68 | 84.81 | 72.80 | 7.33 |
| SliceGPT | 67.52 | 28.20 | 60.61 | 55.41 | 44.15 | 25.34 | 40.70 | 45.99 | 14.87 |
| Shortened-llama | 65.02 | 32.40 | 68.01 | 64.64 | 57.55 | 33.02 | 53.11 | 53.39 | 17.22 |
| ShortGPT | 65.38 | 32.00 | 68.61 | **67.32** | 58.43 | 35.32 | 53.37 | 54.35 | 18.35 |
| MoD | 50.28 | 33.00 | 65.56 | 51.38 | 54.01 | 30.20 | 38.09 | 46.07 | 40.42 |
| D-LLM | 50.00 | 31.80 | 58.54 | 51.78 | 48.30 | 26.88 | 44.82 | 44.59 | 52.78 |
| SkipGPT | 66.57 | **37.60** | 70.78 | 56.75 | **65.17** | **42.66** | **72.39** | 58.85 | 14.35 |
| LoRAP | 70.18 | 34.00 | 71.27 | 61.80 | 59.69 | 36.52 | 62.54 | 56.57 | 16.08 |
| + UniRank | **71.12** | 35.20 | **73.45** | 66.17 | 63.84 | 39.29 | 64.89 | **59.13** | **14.03** |

2.5. At 40% sparsity, it delivers a 2.6% accuracy gain and a 2.0 perplexity reduction. These consistent improvements validate the efficacy of the proposed UniRank framework.

### 4.4 Further Analysis

Table 3: Performance of UniRank as a Plug-and-Play Module with Different Decomposition Methods under 0.4 sparsity.

| **Decomp.** | **Alloca.** | **Zero-Shot** | **Fine-Tuning** | |
|---|---|---|---|---|
| | | WT2↓ | WT2↓ | Rea.↑ |
| Dense | / | / | 8.70 | 69.73% |
| SVD | Uniform | 134.68 | 28.57 | 49.20% |
| +UniRank | S&T | **86.20** | **26.10** | **49.50%** |
| AFM | Uniform | 43.93 | 23.12 | 52.80% |
| +UniRank | S&T | **28.06** | **22.01** | **53.97%** |
| AWSVD | Heuristic | 28.05 | 23.90 | 53.67% |
| +UniRank | S&T | **26.75** | **22.89** | **54.02%** |
| Dobi | Learning | / | 14.10 | 61.10% |
| +UniRank | S&T | / | **11.74** | **63.05%** |

**Generalization across decomposition.** We integrate UniRank with multiple decomposition methods on LLama2-7B. It consistently matches or surpasses each method's original rank assignment, demonstrating strong generalization. This indicates our method effectively estimate both inter- and intra-parameter low-rank structures, yielding a near-optimal allocation. It is worth noting that heuristic manual assignment (reliant on model-specific observations) degrades on Llama3 (AWSVD on Table 2) versus Llama2 (AWSVD on Table 3) because of different attention architectures, our approach adapts automatically to different architectures, sustaining robust performance across models.

**Rank allocation visualization** Our experimental analysis (Figure 4) on rank allocation visualization reveals distinct patterns across methods. The manual approach assigns uniform ranks per layer, heuristically allocating more to `wv`,`wo` matrices, such architecture-dependent heuristic limits generalization, explaining our method's greater gains on Llama3.1-8B. The learning-based method exhibits a conservative strategy, with rank fluctuations despite global optimization. Given the significant

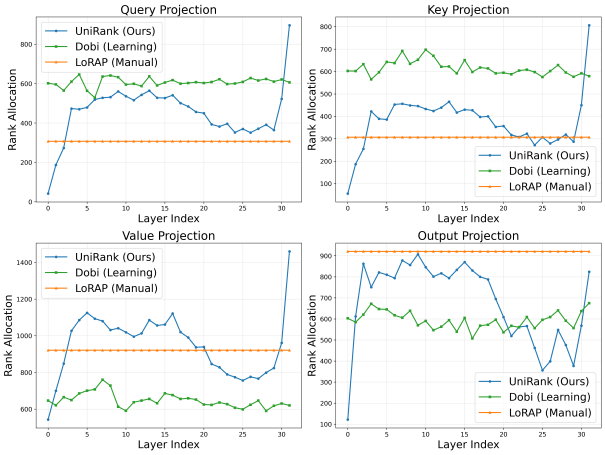


Figure 4: Visualization of rank allocation for Llama2-7b. Our method (UniRank) exhibits notably more aggressive rank allocation with larger inter-layer variance, whereas the learning-based method (Dobi) demonstrates weaker adaptation even with task fine-tuning.

layer-wise variability in low-rank properties, e.g., the singular value proportion for retaining 99% energy in `wq`, `wk` parameters ranges from 10% to 60%, such subtle adaptation is not enough to match the inherent low-rank structure. In contrast, our method shows a more aggressive and sensitive allocation, dynamically adapting to inter-layer and inter-parameter low-rank characteristics, which enhances compression efficacy.

Table 4: Grouping Ablation of UniRank

| Method | Avg. Acc. (↑) | PPL (↓) |
|---|---|---|
| UniRank-parameter | 63.44 | 13.01 |
| UniRank-module | 65.26 | 10.93 |
| UniRank-all | **66.05** | **10.29** |

**Grouping Ablation for Global Singular Component Truncation** We conduct ablation on parameter grouping strategies for global rank allocation. We design three grouping settings: (1) parameter-level grouping, which sorts and truncates homogeneous projection parameters across all layers; (2) module-level grouping, which aggregates all parameters of the same functional module across layers; (3) our unified global setting, where all model parameters are ranked within a single group. Parameters across different groups are allocated rank budgets proportional to their original parameter sizes. Experimental results demonstrate that the unified global grouping consistently outperforms both parameter-level and module-level grouping. Separated grouping restricts cross-layer global singular component scheduling, leading to suboptimal rank allocation and inferior downstream performance. This validates the necessity of our holistic global rank assignment paradigm.

**Hyperparameter Search for Global-Local Metric Balancing** While the base S&T metric, defined in Equation 2 exhibits strong generalization ability, we observe that minimal task-aware adaptation further enhances efficacy. We generalize S&T as:

$$s_{l,i} = (\mathcal{I}_l)^{\alpha} \cdot \frac{\sigma_{l,i}^2}{\|\Sigma_l\|_F^2} \tag{4}$$

where adds $\alpha > 0$ for modulating the trade-off between structural importance and spectral energy concentration.

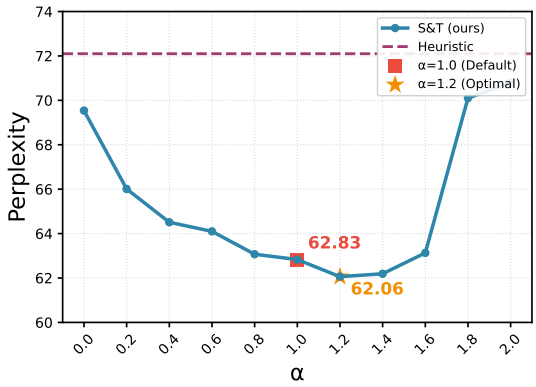


Figure 5: Zero-shot perplexity of S&T with varying $\alpha$. Model: Llama2-7b with 50% sparsity. We choose $\alpha = 1.0$ for simplicity, while grid search finds $\alpha = 1.2$ as optimal (lower perplexity). The dashed line represents maual allocation.

We perform an efficient grid search over $\alpha \in \{0.0, 0.2, 0.4, \cdots, 2.0\}$ using only a tiny validation subset (100 samples from WikiText trainset). As shown in Figure 5 (Llama2-7b with 50% sparsity), selecting $\alpha$ via validation loss can easily improves performance over the default $\alpha = 1.0$ configuration.

This refinement carries three key implications. First, it validates the *compositional interpretability* of S&T: both components encode meaningful signals whose optimal weighting is task-dependent. Second, it broadens UniRank's applicability, i.e., when minimal validation data is available, practitioners gain a near-zero-overhead pathway to boost performance; when unavailable, $\alpha = 1$ remains a strong default. Third, it underscores UniRank's efficiency advantage over learning-based rank allocation methods, which require expensive end-to-end

optimization or gradient estimation. In contrast, a single scalar search suffices to adapt globally computed metrics to local task characteristics.

Table 5: Ablation study of UniRank components, every component contributes positively to the model performance. Model: LLaMA3.1-8B with 25% sparsity. Metrics: average accuracy (Avg. Acc.) on reasoning tasks, perplexity (PPL) on WikiText Best results in **bold**.

| Configuration | Avg. Acc. (↑) | PPL (↓) |
|---|---|---|
| Baseline | 63.09 | 12.82 |
| + S&T(local) | 64.70 | 12.11 |
| + S&T(local + global) | 65.67 | 10.85 |
| + RPFT | **66.05** | **10.29** |

**Ablation studies** We perform ablation experiments to validate the effectiveness of each UniRank component, adopting AWSVD decomposition with uniform rank allocation and standard LoRA fine-tuning as the strong baseline. We progressively integrate three core modules: local S&T relying only on singular energy ratio, full S&T incorporating global functional importance, and rank-preserving fine-tuning (RPFT). Table 8 shows that each component brings steady, cumulative performance gains. S&T(local) outperforms uniform rank allocation by 1.61 average points, proving that singular energy-based rank assignment is more reasonable than uniform assignment. Full S&T further improves the score by 0.93 points over local S&T, verifying that high-importance layers with greater feature divergence require higher rank retention to preserve representation capability. Replacing standard LoRA with RPFT yields the most prominent improvement of 3.0 points against the baseline, attributed to RPFT's strict effecitive-rank constraint.

Table 6: Training cost for compressing Llama2-7b. UniRank incurs negligible training overhead, representing a drastic reduction over learning-based counterparts.

| Method | Decom. | Alloc. | FT | re-Decom. |
|---|---|---|---|---|
| LoRAP | 29 min | 0 (manual) | 7 h | 29 min |
| UniRank | 29 min | 2min | 7 h | 0 (no need) |
| Dobi | 11 h | 220 h | 0 | 0 |
| UniRank | 11 h | 11 h | 0 | 0 |

**Training cost and inference speedup** (1) Training Overhead. We evaluate training costs by comparing UniRank with LoRAP and DoBi on LLaMA2-7b under identical hardware platform (CPU: Intel Xeon w5-2445, GPU: NVIDIA RTX 6000 Ada $\times 1$). For LoRAP, UniRank adds only 2 minutes for rank allocation but saves 29 minutes in fine-tuning by avoiding secondary decomposition. For DoBi, which performs per-sample forward activation decomposition (nsample=256, seq_len=2048), training time scales with samples and epochs. Since DoBi requires 20 epochs for rank learning versus our single epoch, UniRank reduces this phase by 95% (19/20), demonstrating significant efficiency.

Table 7: Inference time (s) across different sequence length. UniRank exhibits identical inference complexity to others, with comparable practical inference acceleration performance.

| Seq_len | Dense | LoRAP | UniRank |
|---|---|---|---|
| 64 | 0.0157 | 0.0128 | 0.0124 |
| 128 | 0.0211 | 0.0143 | 0.0149 |
| 256 | 0.0246 | 0.0172 | 0.0177 |
| 512 | 0.0426 | 0.0258 | 0.0255 |
| 1024 | 0.0840 | 0.0489 | 0.0467 |
| 2048 | 0.1783 | 0.1066 | 0.1045 |

(2) Inference Acceleration. UniRank maintains the total rank count, so theoretical complexity matches baselines. Experimental results on various sequence lengths confirm this: compared to dense and sparse models (50% sparsity), speedup is consistent with minor fluctuations. Thus, our method achieves practical acceleration with performance gains.

## 5 Conclusion

This paper presents an efficient inference framework **UniRank** for LLM via low-rank decomposition. We propose **Sorting-and-Truncation**, which adaptively assigns ranks using a aggregated metric combining global functional importance and local singular energy, eliminating manual limitations and reducing computational overhead. Additionally, we introduce **Rank-Preserving Fine-Tuning** to enable parameter-efficient tuning without rank inflation or re-decomposition. UniRank demonstrates strong versatility by working seamlessly with various existing decomposition methods, whether applied to parameters or activations. Visualization reveals that the proposed method yields more aggressive rank distributions, validating its optimization efficacy. To sum up, UniRank offers a practical and generalizable solution for inference. Future work will extend its principles to even finer-grained compression (e.g. per-parameter importance evaluation) and broader architectural families.

## Limitations

Despite promising practical performance, the proposed method is derived from geometric intuition rather than rigorous theoretical derivation such as error analysis. Moreover, the global functional importance is evaluated based on equivalent transformation of large modules instead of individual parameters. Such asymmetry constrains the performance ceiling. Specifically, trivial parameters may still exist inside critical modules, leaving further pruning potential untapped. We will explore this issue in future work and wish to provide insights for relevant research communities.

## References


Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 225–237.

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. In *The Twelfth International Conference on Learning Representations*.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, and 1 others. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Xiaodong Chen, Yuxuan Hu, Jing Zhang, Yanling Wang, Cuiping Li, and Hong Chen. 2025. Streamlining redundant layers to compress large language models. In *International Conference on Representation Learning*, volume 2025, pages 30362–30383.

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*.

Giuseppe Crupi, Rosalia Tufano, Alejandro Velasco, Antonio Mastropaolo, Denys Poshyvanyk, and Gabriele Bavota. 2025. On the effectiveness of llm-as-a-judge for code generation and summarization. *IEEE Transactions on Software Engineering*.

DeepSeek-AI. 2026. Deepseek-v4: Towards highly efficient million-token context intelligence.

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

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. *Preprint*, arXiv:1909.00512.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: massive language models can be accurately pruned in one-shot. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

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, and 5 others. 2024a. The language model evaluation harness.

Shangqian Gao, Ting Hua, Yen-Chang Hsu, Yilin Shen, and Hongxia Jin. 2024b. Adaptive rank selections for low-rank approximation of language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 227–241. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

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, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Weizhong Huang, Yuxin Zhang, Xiawu Zheng, Fei Chao, and Rongrong Ji. 2025. Determining layer-wise sparsity for large language models through a theoretical perspective. In *Forty-second International Conference on Machine Learning*.

Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. 2014. Speeding up convolutional neural networks with low rank expansions. In *Proceedings of the British Machine Vision Conference*, pages 88–1. British Machine Vision Association.

Ajay Kumar Jaiswal, Yifan Wang, Lu Yin, Shiwei Liu, Runjin Chen, Jiawei Zhao, Ananth Grama, Yuandong Tian, and Zhangyang Wang. 2025. From low rank gradient subspace stabilization to low-rank weights: Observations, theories, and applications. In *International Conference on Machine Learning*, pages 26740–26756. PMLR.

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.

Yikun Jiang, Huanyu Wang, Lei Xie, Hanbin Zhao, Hui Qian, John Lui, and 1 others. 2024. D-llm: A token adaptive computing resource allocation strategy for large language models. *Advances in Neural Information Processing Systems*, 37:1725–1749.

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*.

Guangyan Li, Yongqiang Tang, and Wensheng Zhang. 2024. Lorap: Transformer sub-layers deserve differentiated structured compression for large language models. In *International Conference on Machine Learning*.

Yi Li, Zhichun Guo, Miao Yin, and Bingzhe Li. 2026. Lestd: Llm compression via learning-based sparse tensor decomposition. In *The Fourteenth International Conference on Learning Representations*.

Chi-Heng Lin, Shangqian Gao, James Smith, Abhishek Patel, Shikhar Tuli, Yilin Shen, Hongxia Jin, and Yen-Chang Hsu. 2025a. Modegpt: Modular decomposition for large language model compression. In *International Conference on Learning Representations*, volume 2025, pages 101355–101390.

Xin Lin, Zhenya Huang, Zhiqiang Zhang, Jun Zhou, and Enhong Chen. 2025b. Explore what llm does not know in complex question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24585–24594.

Yu-Chen Lu, Chong-Yan Chen, Chi-Chih Chang, Yu-Fang Hu, and Kai-Chiang Wu. 2025. Flrc: Fine-grained low-rank compressor for efficient llm inference. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 14956–14966.

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. *Preprint*, arXiv:2403.03853.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *ICLR*, volume abs/1609.07843.

Zhendong Mi, Bian Sun, Grace Li Zhang, and Shaoyi Huang. 2025. Layer-wise dynamic rank for compressing large language models. *arXiv preprint arXiv:2509.25622*.

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.

Francesco Maria Molfese, Luca Moroni, Luca Gioffré, Alessandro Scirè, Simone Conia, and Roberto Navigli. 2025. Right answer, wrong score: Uncovering the inconsistencies of llm evaluation in multiple-choice question answering. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 18477–18494.

Wang Qinsi, Jinghan Ke, Masayoshi Tomizuka, Kurt Keutzer, and Chenfeng Xu. 2025. Dobi-SVD: Differentiable SVD for LLM compression and some new perspectives. In *The Thirteenth International Conference on Learning Representations*.

David Raposo, Sam Ritter, Blake Richards, Timothy Lillicrap, Peter Conway Humphreys, and Adam Santoro. 2024. Mixture-of-depths: Dynamically allocating compute in transformer-based language models. *arXiv preprint arXiv:2404.02258*.

Rajarshi Saha, Naomi Sagan, Varun Srivastava, Andrea J Goldsmith, and Mert Pilanci. 2024. Compressing large language models using low rank and low precision decomposition. *Advances in Neural Information Processing Systems*, 37:88981–89018.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Amrith Setlur, Saurabh Garg, Xinyang Geng, Naman Garg, Virginia Smith, and Aviral Kumar. 2024. Rl on incorrect synthetic data scales the efficiency of llm math reasoning by eight-fold. *Advances in Neural Information Processing Systems*, 37:43000–43031.

James Seale Smith, Chi-Heng Lin, Shikhar Tuli, Haris Jeelani, Shangqian Gao, Yilin Shen, Hongxia Jin, and Yen-Chang Hsu. 2025. Flexigpt: Pruning and extending large language models with low-rank weight sharing. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 718–730.

Jiayi Tian, Ryan Solgi, Jinming Lu, Yifan Yang, Hai Li, and Zheng Zhang. 2026. Flat-llm: Fine-grained low-rank activation space transformation for large

language model compression. In *Findings of the Association for Computational Linguistics: EACL 2026*, pages 2988–3002.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, and 1 others. 2023b. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. 2025. Svd-llm: Truncation-aware singular value decomposition for large language model compression. In *International Conference on Learning Representations*, volume 2025, pages 19299–19319.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

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,*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Biao Zhang and Rico Sennrich. 2019. Root mean square layer normalization. *Preprint*, arXiv:1910.07467.

Stephen Zhang and Vardan Papyan. 2025. Oats: Outlier-aware pruning through sparse and low rank decomposition. In *International Conference on Learning Representations*, volume 2025, pages 50870–50895.

Ziyao Zhang, Chong Wang, Yanlin Wang, Ensheng Shi, Yuchi Ma, Wanjun Zhong, Jiachi Chen, Mingzhi Mao, and Zibin Zheng. 2025. Llm hallucinations in practical code generation: Phenomena, mechanism, and mitigation. *Proceedings of the ACM on Software Engineering*, 2(ISSTA):481–503.

Anhao Zhao, Fanghua Ye, Yingqi Fan, Junlong Tong, Jing Xiong, Zhiwei Fei, Hui Su, and Xiaoyu Shen. 2025. SkipGPT: Each token is one of a kind. In *Forty-second International Conference on Machine Learning*.

# A Appendix

## A.1 Baseline Methods

To comprehensively validate the effectiveness of our method, we compare it with representative state-of-the-art techniques covering static model compression and dynamic computation allocation paradigms, which are detailed as follows:

**Low Rank Compression**

- **AFM** (Yu and Wu, 2023): it reveals that Transformer activations, rather than weights, possess intrinsic low-rank structure. And it proposes Atomic Feature Mimicking to aggregate activation statistics for effective low-rank decomposition.
- **LoRAP** (Kim et al., 2024): it adopts differentiated structured compression for Transformers, applying low-rank decomposition to attention modules and channel pruning to MLP modules.
- **Dobi** (Men et al., 2024): it establishes a fully activation-centric compression paradigm, performing differentiable SVD and truncation directly on activations instead of weights. It is worth noting that the DoBi method requires performing SVD decomposition on every linear layer's forward activation using a calibration set. Such heavy computation cannot be supported on our experimental platform. Accordingly, we only apply DoBi to attention modules and exclude MLP modules from compression. This setting is adopted for both the original DoBi and DoBi+unirank to guarantee fair comparisons.

**Static Pruning**

- **SliceGPT** (Ashkboos et al., 2024): it eliminates entire parameter rows and columns by applying PCA to orthogonally transformed model parameters.
- **Shortened-llama** (Kim et al., 2024): it constructs lightweight models by pruning consecutive network layers.
- **ShortGPT** (Men et al., 2024): it adopts Block Influence (BI) to quantitatively assess layer importance and eliminate insignificant layers.

**Dynamic Computation Allocation**

- **Mixture-of-Depths (MoD)** (Raposo et al., 2024): it utilizes a greedy routing strategy and maintains a fixed sparsity ratio for each layer block to dynamically determine the execution or skipping of computational units for individual tokens.

- **D-LLM** (Jiang et al., 2024): it realizes global adaptive sparsity by dynamically adjusting layer-wise computation based on input features
- **SkipGPT** (Zhao et al., 2025): it achieves finer-grained dynamic control by decoupling intra-layer attention and MLP modules.

### A.2 Rank Dynamics and Information Preservation

Let $\mathbf{W}_0 = \mathbf{U}_0\mathbf{V}_0^\top \in \mathbb{R}^{m\times n}$ denote a pre-compressed weight matrix with $\text{rank}(\mathbf{W}_0) = k_{\text{save}}$ ( $\mathbf{U}_0 \in \mathbb{R}^{m\times k_{\text{save}}}$ , $\mathbf{V}_0 \in \mathbb{R}^{n\times k_{\text{save}}}$ ). Consider two adaptation paradigms:

**(i) Standard LoRA**: The adapted weight is $\mathbf{W}_{\text{LoRA}} = \mathbf{W}_0 + \mathbf{A}\mathbf{B}^\top$ , where $\mathbf{A} \in \mathbb{R}^{m\times k_{\text{lora}}}$ , $\mathbf{B} \in \mathbb{R}^{n\times k_{\text{lora}}}$ are randomly initialized. The subsequent re-decomposition to rank $k_{save}$ almost always incurs an approximation error.

Enforcing the original compression budget requires re-decomposition to rank $k_{\text{save}}$ . By the Eckart–Young–Mirsky theorem, the minimal approximation error is

$$\min_{\text{rank}(\mathbf{W}')\le k_{\text{save}}} \|\mathbf{W}_{\text{LoRA}} - \mathbf{W}'\|_F \tag{5}$$

$$= \sqrt{\sum_{i=k_{\text{save}}+1}^{\min(m,n)} \sigma_i^2(\mathbf{W}_{\text{LoRA}})} > 0, \tag{6}$$

where strict positivity follows from $\sigma_{k_{\text{save}}+1}(\mathbf{W}_{\text{LoRA}}) > 0$

**(ii) RPFT**: Partition $\mathbf{U}_0 = [\mathbf{U}_{\text{fix}}, \mathbf{U}_{\text{train}}^{(0)}]$ , $\mathbf{V}_0 = [\mathbf{V}_{\text{fix}}, \mathbf{V}_{\text{train}}^{(0)}]$ with $k_{\text{fix}} + k_{\text{train}} = k_{\text{save}}$ . During fine-tuning, only $\mathbf{U}_{\text{train}}$ , $\mathbf{V}_{\text{train}}$ are updated, yielding

$$\mathbf{W}_{\text{RPFT}}^{(t)} = \underbrace{[\mathbf{U}_{\text{fix}}, \mathbf{U}_{\text{train}}^{(t)}]}_{\mathbf{U}^{(t)}\in\mathbb{R}^{m\times k_{\text{save}}}} \underbrace{[\mathbf{V}_{\text{fix}}, \mathbf{V}_{\text{train}}^{(t)}]^\top}_{(\mathbf{V}^{(t)})^\top\in\mathbb{R}^{k_{\text{save}}\times n}} . \tag{7}$$

Critically, $\text{rank}(\mathbf{W}_{\text{RPFT}}^{(t)}) \le \min\{\text{rank}(\mathbf{U}^{(t)}), \text{rank}(\mathbf{V}^{(t)})\} \le r_{\text{save}}$ holds *by construction* for all $t$ , irrespective of optimization trajectory or subspace alignment. The adapted weights remain exactly representable within the pre-allocated rank budget—no re-decomposition is required, and all task-specific information is preserved without projection-induced distortion.

### A.3 Attention-MLP Decoupling

We further decouple each Transformer layer into individual attention and MLP modules, both adhering to the residual structure for similarity-based rank allocation. Statistically, the attention module yields a higher average input-output cosine similarity (94%) than the MLP module (89%), indicating a lower effective intrinsic rank and better low-rank compressibility. This aligns with LoRAP observations and validates our similarity-driven rank assignment paradigm. The marginal overall improvement (i.e. perplexity reduce 0.4) stems from slight parameter budget reallocation toward MLP after decoupling. Quantitatively, the divergent singular components between layer-wise and module-wise allocation only account for 2.1% of the total, leading to negligible performance discrepancy.

Table 8: Ablation study of UniRank components, every component contributes positively to the model performance. Model: LLaMA3.1-8B with 25% sparsity. Metrics: average accuracy (Avg. Acc.) on reasoning tasks, perplexity (PPL) on WikiText Best results in **bold**.

| Method | Avg. Acc. (↑) | PPL (↓) |
|---|---|---|
| UniRank-all | 66.05 | 10.29 |
| UniRank-decouple | 66.02 | 9.88 |