TASK RESIDUALS ARE LOW-RANK: EFFICIENT PRUNING AND RECOVERY OF LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Structured pruning reduces the inference cost of large language models by removing complete architectural units. However, existing methods either rely on task-agnostic importance estimates that overlook task-specific structural requirements or repeatedly evaluate all candidate structural blocks for each downstream task, resulting in substantial adaptation overhead. Through a systematic block-wise masking study, we empirically identify a previously overlooked structure in block importance: it can be decomposed into a generic component shared across tasks and task-specific residuals concentrated in a compact low-rank subspace. Building on this finding, we propose ReTaPER, a residual subspace-guided framework for task-aware pruning with efficient recovery. ReTaPER first learns a low-rank residual basis from block-masking responses. For a new task, it probes only a small set of informative anchor blocks, reconstructs the complete task residual within the learned subspace, and combines it with generic importance to derive a task-specific pruning order. After pruning, the reconstructed residual further guides the allocation of a fixed LoRA rank budget, assigning greater adaptation capacity to retained blocks that are more sensitive to the target task. Extensive experiments across nine tasks demonstrate that ReTaPER achieves state-of-the-art pruning performance both with and without recovery. Code is available at `https://anonymous.4open.science/r/ReTaPER-ICLR2027-4DD3/`.

1 INTRODUCTION

Large language models (LLMs) deliver strong performance across tasks, but their memory and computational costs hinder deployment in resource-constrained and latency-sensitive settings (Kwon et al., 2023). Model pruning reduces these costs by eliminating redundant parameters or components (Ma et al., 2023; Frantar & Alistarh, 2023). Among pruning approaches, structured pruning removes complete architectural units while preserving dense computation, making compressed models amenable to practical acceleration (An et al., 2024; Ashkboos et al., 2024).

Many structured pruning methods estimate the importance of candidate structural units from generic calibration data and reuse a fixed pruned architecture across tasks (Song et al., 2024; Zhong et al., 2025). Although efficient, this strategy overlooks task-dependent computational requirements. Recent task-aware methods instead derive a separate pruning plan for each downstream task from early fine-tuning gradients (Huang et al., 2026), intermediate output distributions (Jo et al., 2026), or task-level candidate evaluations (Naim et al., 2026). Although better tailored to the target task, they repeat structural-importance estimation for every new task, incurring substantial adaptation overhead.

We therefore explore whether structural importance exhibits reusable patterns across tasks. A block-wise masking study over the 57 subjects of the MMLU dataset (Hendrycks et al., 2021) reveals two patterns. First, block importance contains a strong generic trend, with some blocks consistently important and others broadly redundant. Second, after removing this trend, related subjects exhibit coherent residual patterns, while the residuals across subjects concentrate in a compact low-rank subspace.

Based on this insight, we propose ReTaPER, a residual subspace-guided framework for efficient task-aware pruning with recovery. In an offline stage, ReTaPER uses singular value decomposition (SVD) to learn a compact basis from MMLU subject residuals and selects a small set of anchor blocks that cover the learned directions. For a new task, the online stage probes only these anchors,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

reconstructs the residual over all blocks, and combines it with generic importance to produce a task-specific pruning order without a full structural sweep. Because retained blocks differ in task sensitivity, ReTaPER further reuses the reconstructed residual to distribute a fixed LoRA rank budget across retained blocks, allowing the same task signal to guide both pruning and recovery.

Our main contributions are summarized as follows:

- Our empirical study reveals that task-specific block importance decomposes into a generic component shared across tasks and task residuals concentrated in a low-rank subspace.
- Building on this finding, we propose ReTaPER, which reconstructs task residuals from a small set of anchor probes to derive task-specific pruning orders without evaluating every block.
- We further reuse the residual signal for dynamic LoRA rank allocation, concentrating a fixed adaptation budget according to the task sensitivity of retained blocks.
- Experiments on nine downstream tasks and two model backbones show that ReTaPER consistently outperforms representative baselines under both pruning-only and recovery settings.

2 Related Work

Structured pruning. Existing methods span multiple structural granularities. At the width level, LLM-Pruner uses gradient-based importance to prune dependency-coupled attention heads and MLP channels, FLAP removes input channels according to activation fluctuation, and SliceGPT applies PCA-based rotations and matrix slicing to reduce the model's hidden dimension (Ma et al., 2023; An et al., 2024; Ashkboos et al., 2024). At the block level, BlockPruner and EntroDrop treat each MHA or MLP module as a pruning unit, ranking blocks by perplexity impact (Zhong et al., 2025) or representation entropy (Yang et al., 2026). At the coarsest level, layer-pruning methods remove complete Transformer layers according to iterative calibration perplexity (Song et al., 2024), input–output similarity (Men et al., 2025), or parameter merging (Yang et al., 2024). Despite their different pruning units, these methods derive a fixed structure from generic calibration data, leaving task-dependent structural requirements unmodeled.

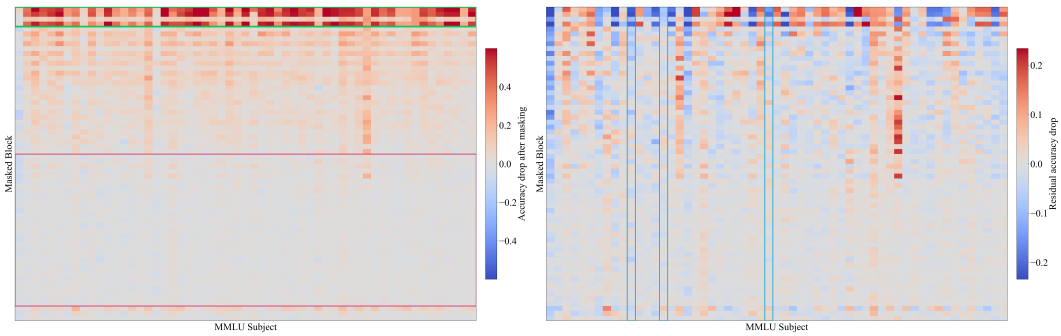
Task-aware and adaptive pruning. Task awareness has been introduced at both the input and task levels. Among input-conditioned approaches, IG-Pruning discovers candidate layer masks and routes each input to a suitable mask (Qiao et al., 2025), while Instruction-Following Pruning trains an instruction-conditioned predictor for sparse FFN masks (Hou et al., 2025). Probe Pruning instead uses partial hidden-state look-ahead to select channels for each input batch (Le et al., 2025). At the task level, GradPruner accumulates early fine-tuning gradients to rank layers (Huang et al., 2026), TaBP estimates layer importance from intermediate output distributions (Jo et al., 2026), and TELLTALE iteratively tests layer removal against task accuracy (Naim et al., 2026). These methods tailor pruning to the current input or task without transferring importance information across tasks. GISP pools multi-task calibration data to optimize a single shared subnetwork (Wang et al., 2026), whereas Tian et al. combine generic and task-specific importance using activation-based grouping (Tian et al., 2025). Both incorporate shared information, but deriving a task-specific structure for a new task still requires importance evaluation across the full model. ReTaPER instead learns a reusable low-rank model of cross-task variation and reconstructs new-task importance from only a few anchor probes.

3 Problem Formulation and Observations

3.1 Problem Formulation

We study block-level structural pruning of a pretrained large language model, treating each MHA or MLP module in a Transformer layer as a removable block. The model is thus viewed as a stack of N removable MHA/MLP blocks, denoted by $\mathcal{B} = \{b_i\}_{i=1}^{N}$.

Given a downstream task $t \in \mathcal{T}$ and a target parameter-pruning ratio α, we associate each block b_i with a task-specific structural importance $d_{i,t}$, where smaller values indicate safer removal. We



(a) Accuracy drop (b) Residual drop

Figure 1: MMLU block-masking responses for Llama-3.1-8B. (a) Accuracy drops from masking each of the 64 MHA/MLP blocks. (b) Residual drops after subtracting the generic component. Colored boxes mark patterns discussed in the text.

define the task-specific pruning order as

$$\pi_t = \text{argsort}_{i \in \{1,\dots,N\}} d_{i,t}. \tag{1}$$

Following π_t, we bypass blocks until their cumulative parameter ratio reaches α. For each bypassed block, its MHA or MLP transformation is disabled and its input is passed unchanged to the next model component; all other components remain unchanged. The core problem is therefore estimating $d_{i,t}$ accurately.

3.2 Motivating Observations

Our method is motivated by a block-wise masking study on MMLU. MMLU is a broad benchmark comprising 57 subjects spanning STEM, the humanities, and professional fields. For each MMLU subject, we bypass each block in turn and measure the resulting accuracy drop relative to the full model. Let $\mathcal{S} = \{s_j\}_{j=1}^{J}$ be the subject set, with $J = 57$ in MMLU. For subject s_j, let a_j^0 be the full-model accuracy, and a_{ij} be the accuracy after removing block b_i. We collect these drops in the MMLU block-drop matrix $M = [m_{ij}] \in \mathbb{R}^{N \times J}$, where $m_{ij} = a_j^0 - a_{ij}$. Positive values indicate accuracy loss after masking, whereas negative values indicate improved accuracy. As shown in Figure 1(a), the response matrix for Llama-3.1-8B (Grattafiori et al., 2024) is highly structured rather than uniformly noisy. The most visible pattern is a strong depth-wise generic trend: the earliest blocks highlighted by the green box, such as mha@0, mlp@0, and mlp@1, produce large positive drops for most subjects, whereas many deeper blocks highlighted by the purple box produce near-zero or negative drops across subjects. This contrast reveals a task-agnostic component of structural importance: some blocks are broadly unsafe to prune, while others are broadly redundant under the MMLU masking criterion. This leads to our first observation.

Observation 1 (Shared generic importance). *Block-level structural importance contains a generic component that is stable across tasks.*

Motivated by this observation, we decompose block importance $d_{i,t}$ as

$$d_{i,t} = g_i + r_{i,t}, \qquad \mathbf{r}_t = [r_{1,t}, \dots, r_{N,t}]^\top. \tag{2}$$

Here $g_i = \frac{1}{J}\sum_{j=1}^{J} m_{ij}$ captures task-agnostic importance shared across tasks. The residual $\mathbf{r}_t$ captures how task t deviates from this shared pruning trend.

To isolate task-specific deviations, we remove the block-wise generic component g_i for each block b_i, yielding the residual matrix $R = [r_{ij}]$ with $r_{ij} = m_{ij} - g_i$, as visualized in Figure 1(b). After this removal, the dominant row-wise trend is suppressed, while column-wise subject structure remains. For example, college physics, high school physics, and electrical engineering (the blue box) form a coherent residual group: the pairwise mean cosine similarity between their residual response vectors is 0.62, whereas the average cosine similarity between these three subjects and all remaining subjects is only -0.13. This contrast shows that task-specific residual vectors vary substantially

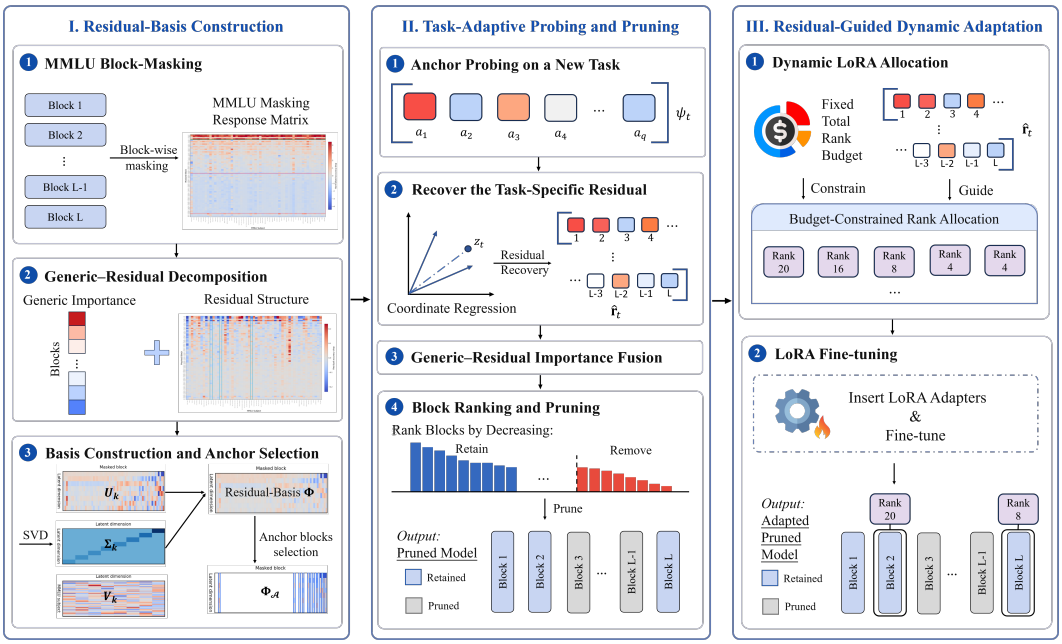


Figure 2: Overview of ReTaPER. The offline stage learns a low-rank residual basis from MMLU block-masking responses; the online stage reconstructs task residuals from anchor probes to guide pruning and dynamic LoRA rank allocation.

across subjects, yet their variation is not arbitrary: related subjects exhibit structured correlations. A natural hypothesis is therefore that these residual vectors can be approximated by a small number of shared basis directions. The SVD-based analysis in Section 5.3 validates this hypothesis, leading to our second observation.

Observation 2 (Low-rank task residuals). *After removing the generic component, task-specific block-importance residuals exhibit structured cross-task correlations and can be approximated in a compact low-rank subspace.*

Cross-backbone evidence for these observations comes from repeating the MMLU analysis on Qwen3-8B (Yang et al., 2025) (Appendix A). Cross-dataset evidence comes from constructing and evaluating a residual basis using 18 selected BIG-bench reference tasks (Srivastava et al., 2023) on Llama-3.1-8B (Appendix B).

4 METHOD

Based on these observations, we propose ReTaPER, which turns the decomposition $d_{i,t} = g_i + r_{i,t}$ into a practical pruning pipeline. As illustrated in Figure 2, ReTaPER comprises a reusable offline stage and a task-specific online stage. In the offline stage, ReTaPER constructs the MMLU block-masking response matrix, learns a compact residual basis, and selects a small set of anchor blocks that cover the learned residual directions. In the online stage, it probes only these anchors for each new downstream task, reconstructs the residual over all blocks, and combines it with generic importance to derive a task-specific pruning order. After pruning, the same residual signal further guides the allocation of a fixed LoRA rank budget over the retained blocks, thereby mitigating the performance degradation introduced by structural pruning. See Appendix C for the full procedure.

4.1 RESIDUAL-BASIS CONSTRUCTION

The residual matrix R defined above captures how each MMLU subject deviates from the generic pruning trend. Before factorization, we form $\widetilde{R} = [\widetilde{r}_{ij}]$ by centering each subject column and nor-

malizing its scale:

$$\widetilde{r}_{ij} = \frac{r_{ij} - \mu_j}{\sigma_j}. \tag{3}$$

Here μ_j and σ_j denote the mean and standard deviation of the j-th residual column, respectively. This normalization prevents subject-specific offsets or scales from dominating the shared structure.

We then apply rank-k SVD to extract a compact set of block-level residual directions:

$$\widetilde{R} \approx U_k \Sigma_k V_k^\top, \qquad \Phi = U_k \Sigma_k. \tag{4}$$

Here k denotes the retained SVD rank and thus the dimensionality of the learned residual subspace. The matrices $U_k \in \mathbb{R}^{N\times k}$, $\Sigma_k \in \mathbb{R}^{k\times k}$, and $V_k \in \mathbb{R}^{J\times k}$ encode block-side directions, their strengths, and subject-side coordinates, respectively. We use $\Phi \in \mathbb{R}^{N\times k}$ as the residual basis.

For a new downstream task t, evaluating the drop of every removable block would defeat the purpose of fast task-adaptive pruning. ReTaPER therefore probes only a small set of anchor blocks. We select an anchor index set $\mathcal{A} = \{a_\ell\}_{\ell=1}^{q}$, where q is the number of anchor probes, and let $\Phi_{\mathcal{A}} \in \mathbb{R}^{q\times k}$ contain the corresponding rows of Φ. A useful anchor set should cover the learned residual directions while avoiding redundant probes. Let $\mathfrak{A}_q$ denote the collection of all q-anchor subsets. We use the following objective to guide anchor selection:

$$\arg\max_{\mathcal{A}\in\mathfrak{A}_q} \log\det\left(\Phi_{\mathcal{A}}^\top \Phi_{\mathcal{A}} + \epsilon I_k\right). \tag{5}$$

Here I_k is the $k \times k$ identity matrix, and $\epsilon > 0$ ensures nonsingularity. The objective promotes a well-conditioned $\Phi_{\mathcal{A}}$. In practice, we first select rows by their components orthogonal to the span of the selected rows until the k-dimensional subspace is covered, then fill the remaining anchors with the largest-norm unselected rows.

4.2 Task-Adaptive Probing and Pruning

For a new task, we randomly sample a small probe set from its training split. We then evaluate the full model and each anchor-masked variant on this set to estimate the q anchor-wise accuracy drops. As in offline preprocessing, we subtract the generic component and normalize the residuals using statistics from the q observed anchors. The resulting values form the probe vector $\boldsymbol{\psi}_t \in \mathbb{R}^q$. Using the corresponding residual-basis submatrix $\Phi_{\mathcal{A}}$, we recover the task-coordinate vector $\mathbf{z}_t \in \mathbb{R}^k$ via ridge regression:

$$\mathbf{z}_t = \arg\min_{\mathbf{z}} \|\Phi_{\mathcal{A}}\mathbf{z} - \boldsymbol{\psi}_t\|_2^2 + \eta\|\mathbf{z}\|_2^2. \tag{6}$$

The recovered residual over all blocks is

$$\widehat{\mathbf{r}}_t = \mathrm{Denorm}_t(\Phi\mathbf{z}_t). \tag{7}$$

Here $\mathrm{Denorm}_t(\cdot)$ reverses the normalization in Eq. (3) using the anchor-estimated statistics.

Finally, we form the importance estimate as $\widehat{d}_{i,t} = g_i + \widehat{r}_{i,t}$, where $\widehat{r}_{i,t}$ is the i-th entry of $\widehat{\mathbf{r}}_t$. We rank blocks by increasing $\widehat{d}_{i,t}$ and prune in this order until the parameter-pruning ratio reaches α, yielding the retained block set $\mathcal{R}_t \subseteq \mathcal{B}$.

4.3 Residual-Guided Dynamic Adaptation

We apply LoRA fine-tuning to recover the performance of the pruned model (Hu et al., 2022), following related uses of LoRA in structured pruning (Ma et al., 2023; Zhang et al., 2024). Conventional LoRA configurations assign the same rank to every adapted block, even though pruning affects different parts of the model unequally. Adaptive-rank methods instead redistribute capacity according to adapter importance estimated during fine-tuning (Zhang et al., 2023; Mao et al., 2024); however, they are designed for general model adaptation rather than pruning-induced recovery. We reuse the reconstructed task residual $\widehat{r}_{i,t}$ as a readily available signal to allocate more capacity to task-sensitive retained blocks while keeping the total rank budget fixed.

We min–max normalize $\widehat{r}_{i,t}$ over the retained blocks to obtain $\bar{r}_{i,t} \in [0,1]$. Let $u_{i,t} \in \mathbb{Z}_{\geq 0}$ denote the rank units assigned to retained block b_i, B_t^{rank} the fixed rank-unit budget for task t, and $\gamma > 0$

the allocation-concentration exponent. We allocate rank units under a diminishing-return objective:

$$\begin{aligned} \max_{\{u_{i,t}\}} \quad & \sum_{b_i \in \mathcal{R}_t} (\bar{r}_{i,t})^{\gamma} \log(1 + u_{i,t}) \\ \text{s.t.} \quad & \sum_{b_i \in \mathcal{R}_t} u_{i,t} = B_t^{\text{rank}}, \qquad u_{i,t} \in \mathbb{Z}_{\geq 0}. \end{aligned} \tag{8}$$

Because the objective is separable and exhibits diminishing returns, we solve it greedily. We initialize $u_{i,t} = 0$ for all retained blocks. At each step, one rank unit is assigned to the block with the largest marginal gain, $(\bar{r}_{i,t})^{\gamma} \log((u_{i,t} + 2)/(u_{i,t} + 1))$, until the budget is exhausted. If each rank unit has size κ, the final LoRA rank of block b_i is

$$\text{rank}_{i,t} = \kappa u_{i,t}. \tag{9}$$

Only retained blocks with $u_{i,t} > 0$ receive LoRA adapters, and all selected adapters are subsequently fine-tuned on the downstream task.

5 Experiments

5.1 Experimental Setup

We report the main results on both Llama-3.1-8B and Qwen3-8B; unless otherwise stated, the remaining experiments use Llama-3.1-8B. We construct the MMLU block-drop matrix using at most 500 examples per subject, bf16 model weights, and batch size 4. For task-residual pruning, we set the SVD rank to $k = 8$ and select $q = 12$ anchor blocks. Each downstream probe set contains at most 200 examples sampled from the training split and excluded from LoRA fine-tuning. The probe and evaluation batch sizes are both 1, the ridge coefficient is $\eta = 10^{-3}$, and the target parameter-pruning ratio is $\alpha = 0.25$. For task-wise adaptation, we use at most 2500 training examples, one epoch, per-device batch size 8, learning rate 2×10^{-4}, LoRA scaling parameter 16, and LoRA dropout 0.05. Dynamic rank allocation uses rank-unit size $\kappa = 4$, budget $B_t^{\text{rank}} = |\mathcal{R}_t|$, and exponent $\gamma = 0.4$. All experiments were conducted on NVIDIA L40 GPUs with 48 GB of memory.

Table 1: Accuracy comparison on Llama-3.1-8B and Qwen3-8B. Within each backbone and fine-tuning setting, the best and second-best results among the pruned models are shown in bold and underlined, respectively. FT denotes downstream fine-tuning.

Method	PubM.	MedM.	LICQ	FIQA-SA	PIQA	Hella.	Wino.	ARC-E	ARC-C	Avg.
Llama-3.1-8B (No FT)										
Base Model	74.80	38.25	41.50	86.32	80.96	79.25	74.19	79.67	55.29	67.80
EntroDrop	54.00	22.28	41.67	<u>62.82</u>	50.82	24.79	50.67	27.02	26.96	40.11
ShortGPT	46.00	<u>24.74</u>	<u>46.17</u>	60.26	<u>64.64</u>	41.69	60.69	<u>44.36</u>	<u>34.90</u>	47.05
IG-Pruning	<u>66.80</u>	**29.98**	43.58	46.58	**73.22**	**65.39**	<u>61.09</u>	**60.20**	34.47	<u>53.48</u>
ReTaPER	**74.00**	23.86	**57.75**	**89.74**	64.47	<u>53.86</u>	**62.27**	44.07	**35.32**	**56.15**
Llama-3.1-8B (FT)										
Tuned Base Model	77.80	59.57	65.58	91.45	82.59	79.33	77.51	86.62	61.77	75.80
LLM-Streamline	**75.80**	33.33	44.75	87.18	69.80	<u>59.66</u>	**72.53**	60.86	39.51	60.38
GradPruner	69.40	<u>43.05</u>	<u>51.08</u>	<u>87.76</u>	<u>70.57</u>	57.67	64.90	<u>61.52</u>	<u>40.37</u>	<u>60.70</u>
ReTaPER	<u>69.60</u>	**55.06**	**67.17**	**91.88**	**73.18**	**59.98**	<u>69.14</u>	**74.49**	**42.24**	**66.97**
Qwen3-8B (No FT)										
Base Model	74.00	50.59	50.17	50.43	77.80	74.93	68.11	80.85	56.57	64.83
EntroDrop	54.60	23.52	41.50	<u>63.11</u>	50.70	25.36	54.87	28.32	28.13	41.12
ShortGPT	52.00	<u>27.54</u>	<u>46.17</u>	57.26	62.02	45.66	56.75	41.58	30.89	46.65
IG-Pruning	<u>56.80</u>	26.80	44.92	45.71	**72.23**	**57.77**	<u>58.67</u>	**59.49**	<u>34.38</u>	<u>50.75</u>
ReTaPER	**64.20**	**30.61**	**54.17**	**66.38**	<u>69.21</u>	<u>51.22</u>	**66.96**	<u>47.23</u>	**41.45**	**54.60**
Qwen3-8B (FT)										
Tuned Base Model	74.60	60.00	73.33	88.03	80.85	74.50	70.78	86.87	65.02	74.89
LLM-Streamline	55.40	34.85	59.17	91.63	<u>73.02</u>	**52.43**	<u>65.80</u>	57.57	<u>40.05</u>	<u>58.88</u>
GradPruner	<u>57.80</u>	<u>42.89</u>	<u>61.80</u>	<u>91.75</u>	**73.93**	51.23	50.72	<u>57.83</u>	39.21	58.57
ReTaPER	**60.20**	**53.29**	**76.17**	**94.32**	70.45	<u>51.92</u>	**67.27**	**65.35**	**41.84**	**64.53**

Downstream pruning is evaluated on nine tasks spanning specialized domains and general-purpose evaluation: PubMedQA (Jin et al., 2019) and MedMCQA (Pal et al., 2022) for medical question answering, LegalBench International Citizenship Questions (LICQ) (Guha et al., 2023) for legal reasoning, FIQA-SA (Maia et al., 2018) for financial sentiment analysis, and PIQA (Bisk et al., 2020), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2020), ARC-Easy, and ARC-Challenge (Clark et al., 2018) as widely used general-purpose benchmarks.

We compare ReTaPER with five representative baselines spanning task-agnostic, input-adaptive, and task-specific pruning. EntroDrop (Yang et al., 2026) and ShortGPT (Men et al., 2025) derive task-agnostic pruning plans, while LLM-Streamline additionally performs post-pruning recovery (Chen et al., 2025). IG-Pruning selects pruning masks according to individual inputs (Qiao et al., 2025), whereas GradPruner derives a separate pruning plan for each downstream task (Huang et al., 2026). ReTaPER couples task-specific pruning with fixed-budget dynamic LoRA adaptation. For reference, we report both the original Base Model and a Tuned Base Model adapted with ReTaPER's dynamic LoRA scheme. More detailed experimental settings are reported in Appendix D.

5.2 Main Results

Table 1 reports results on two model families with and without downstream fine-tuning. On Llama-3.1-8B without fine-tuning, ReTaPER achieves the highest average accuracy among the pruned models (56.15), exceeding IG-Pruning by 2.67 points. It leads on five tasks: three of the four specialized-domain tasks (PubMedQA, LICQ, and FIQA-SA) and two general-purpose tasks (WinoGrande and ARC-Challenge). With fine-tuning, ReTaPER reaches 66.97, outperforming GradPruner and LLM-Streamline by 6.27 and 6.59 points, respectively, and leading on seven tasks. It also exceeds GradPruner, which combines task-specific pruning with post-pruning fine-tuning, on all nine tasks. The Tuned Base Model reaches 75.80, leaving an 8.83-point gap to ReTaPER. Together, these results demonstrate that ReTaPER remains effective across both specialized-domain and general-purpose tasks.

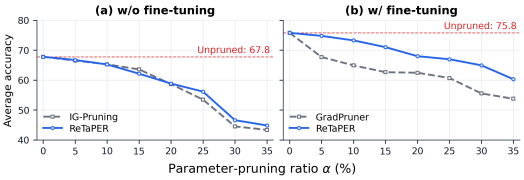


Figure 3: Nine-task average accuracy versus pruning ratio: (a) without and (b) with fine-tuning. Red dashed lines denote unpruned accuracy (67.8 and 75.8).

On Qwen3-8B, ReTaPER leads the pruned-model averages without and with fine-tuning (54.60 and 64.53), exceeding the strongest corresponding baselines by 3.85 and 5.65 points. It leads on six tasks without fine-tuning and seven after fine-tuning. The consistent gains across both backbones support the generality of task-residual modeling beyond a single model family.

Beyond the default pruning ratio, Figure 3 evaluates robustness across compression levels. Without fine-tuning, ReTaPER closely tracks IG-Pruning through 20% pruning and leads by 1.47–2.67 points at ratios of 25–35%. With fine-tuning, it leads GradPruner by 5.54–9.40 points at every nonzero ratio and retains 60.29 versus 53.76 at 35% pruning.

5.3 Validation of the Low-Rank Residual Basis

We examine whether the MMLU residual matrix has a concentrated singular value spectrum. Figure 4 reports the singular values and cumulative explained-energy ratio of the residual matrix used to construct the task-residual basis. The spectrum is sharply concentrated: the first singular direction explains 51.75% of the residual energy, the top eight directions explain 85.15%, and the top twelve directions explain 90.37%. The top sixteen directions cover 93.59% of the residual energy. This compact spectrum supports ReTaPER's low-rank residual assumption and

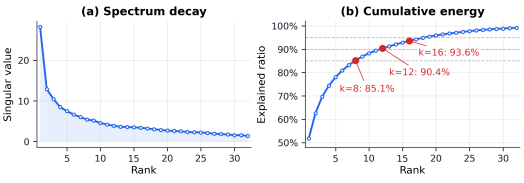


Figure 4: Residual spectrum: (a) singular values and (b) cumulative energy. Red dots mark $k = 8, 12, 16$.

motivates using an SVD basis instead of estimating an independent block-importance vector for each downstream task.

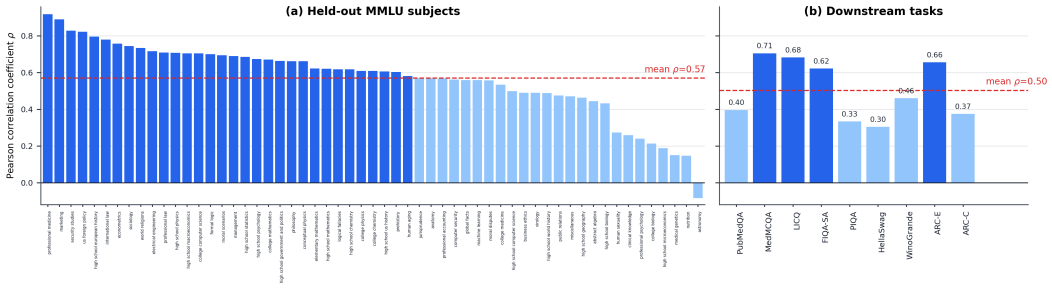


Figure 5: Residual reconstruction with the MMLU-derived basis. (a) Pearson correlations between reconstructed and true residual vectors in leave-one-subject-out MMLU evaluation. (b) Correlations on nine downstream tasks using the basis fitted on all MMLU subjects. Bars in (a) are sorted by ρ; red dashed lines denote panel means.

Having verified the spectral concentration of task residuals, we next examine whether the MMLU-derived basis can reconstruct residual patterns for unseen tasks from sparse anchor observations. We first conduct leave-one-subject-out validation on MMLU: each subject is held out, the basis is fitted to the remaining subjects, and ridge regression reconstructs its residual from the anchor observations. We then fit the basis on all MMLU subjects and apply the same reconstruction procedure to the nine downstream tasks. As shown in Figure 5, the held-out MMLU subjects achieve a mean correlation of $\rho = 0.57$. On the downstream tasks, the MMLU-fitted basis obtains a mean of $\rho = 0.50$, with correlations above 0.60 on MedMCQA, LICQ, FIQA-SA, and ARC-Easy, while reconstruction is weaker on PIQA and HellaSwag. These results indicate that the learned residual structure transfers beyond its reference subjects, although reconstruction quality remains task dependent.

5.4 Ablation and Efficiency Analysis

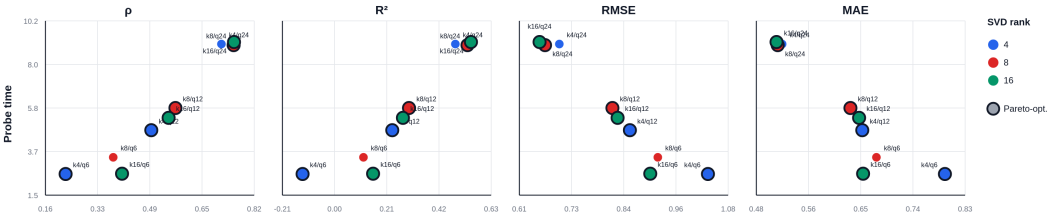


Figure 6: Reconstruction quality versus average probe time over held-out MMLU subjects for different SVD ranks k and anchor counts q. Panels report Pearson ρ, R^2, RMSE, and MAE; black outlines mark panel-wise Pareto optima.

Figure 6 examines the effects of the SVD rank and anchor count on reconstruction quality and probing cost. The best grid configuration, $k = 16$ and $q = 24$, achieves a mean Pearson correlation of $\rho = 0.755$, a mean coefficient of determination $R^2 = 0.551$, a mean RMSE of 0.653, and a mean MAE of 0.510. Increasing the anchor count from $q = 6$ to $q = 12$ consistently improves all four reconstruction metrics, whereas $q = 24$ provides further gains at a probe time of roughly nine minutes per task. Increasing the rank has a weaker and less uniform effect: $k = 16$ is best on all four metrics with $q = 24$. With only $q = 12$, however, increasing k from 8 to 16 slightly degrades reconstruction, suggesting that the available anchors may be insufficient to estimate the higher-dimensional task coordinates reliably. Combined with the spectral elbow near $k = 8$ in Figure 4, these results motivate $k = 8$ and $q = 12$ as the default trade-off between reconstruction quality and probing cost.

Table 2 examines the importance signals used in block selection and LoRA rank allocation. In panel (a), normalized entropy measures how evenly a signal is distributed across blocks, while cosine similarity and Spearman ρ_s measure vector and ranking similarity to the generic signal, respectively. The residual is more diffuse and induces a substantially different pruning order, yet performs poorly when used alone; combining it with the generic signal preserves the stable global ranking and yields the highest average accuracy. In panel (b), uniform all-block adaptation evenly distributes the LoRA rank budget across all 43 retained blocks, whereas the other variants use different importance signals to allocate the same budget dynamically. Residual guidance adapts 40.11 blocks per task on average and achieves the highest accuracy while using slightly less training time and storage than uniform allocation. These results show that the combined signal is better suited to pruning, whereas the residual signal is more effective for allocating recovery capacity.

Table 2: Effects of importance signals on pruning and recovery. Ent., Cos., and ρ_s are normalized entropy, cosine similarity, and Spearman correlation to the generic signal. Time and storage are nine-task totals in minutes and MiB.

(a) Pruning-stage block selection

	Generic	Residual	Combined
Ent.	0.7769	0.9435	0.8039
Cos.	1.0000	0.3589	0.9815
ρ_s	1.0000	0.3474	0.9719
Avg. acc. ↑	52.67	35.12	**56.15**

(b) LoRA-stage rank allocation

	Uniform	Combined	Residual	Generic
Blocks	43.00	30.78	40.11	29.00
Time ↓	11.50	10.51	11.31	**10.25**
Storage ↓	247.52	252.92	**244.69**	255.96
Avg. acc. ↑	64.05	64.13	**66.97**	63.70

Finally, Table 3 compares the computational efficiency of ReTaPER and the baselines. Offline time denotes target-task-independent preprocessing performed once per backbone, whereas online time measures the total task-specific pruning and recovery cost over the nine downstream tasks. The dense model requires 17.4 minutes for evaluation, compared with approximately 13.8 minutes for the pruned models. Because the pruned models have similar evaluation times, this cost is omitted from the table.

Table 3: Efficiency and final accuracy on Llama-3.1-8B. Off. and On. are minutes (online total over nine tasks); memory is GiB. A dash means no corresponding stage.

Method	Off. ↓	On. ↓	GiB ↓	Acc. ↑
EntroDrop	**2.10**	–	**15.06**	40.11
ShortGPT	3.60	–	15.09	47.05
LLM-Streamline	173.10	–	25.28	60.38
GradPruner	–	163.80	35.59	60.70
IG-Pruning	145.20	122.20	19.88	53.48
ReTaPER	202.60	**63.89**	16.10	**66.97**

EntroDrop and ShortGPT incur little preprocessing cost but achieve average accuracies of only 40.11 and 47.05. ReTaPER obtains the highest final accuracy of 66.97, although constructing its reusable MMLU response matrix and residual basis requires the largest offline cost of 202.60 minutes. In contrast, its online cost totals 63.89 minutes, or 7.10 minutes per task, which is 47.7% lower than IG-Pruning and 61.0% lower than GradPruner. ReTaPER also limits peak pruning memory to 16.10 GiB. Overall, it trades a higher one-time offline cost for efficient task-specific pruning and recovery.

Additional analyses report multi-seed robustness (Appendix E), residual-subspace structure and null controls (Appendix F), and the accuracy–efficiency trade-off of sparse probing (Appendix G).

6 Conclusion

Our empirical block-wise masking study reveals that task-specific structural importance decomposes into a shared generic component and task residuals with compact low-rank structure. Building on this insight, ReTaPER exploits a reusable residual representation for task-aware pruning and post-pruning adaptation. Experiments across nine tasks and two model backbones show that ReTaPER outperforms representative pruning and recovery baselines while reducing task-specific online cost. More broadly, our results point toward a task-aware compression paradigm that amortizes structural discovery across tasks through a reusable prior and lightweight task-specific localization. Further discussion is provided in Appendix H.

AI USE STATEMENT

Generative AI tools were used to polish the writing and presentation of this manuscript, including language editing, LaTeX formatting, and figure styling. The tools were not used to conduct experiments or generate results. The authors reviewed the AI-assisted changes and take full responsibility for the final content.

REPRODUCIBILITY STATEMENT

Section 4 and Appendix C describe the method and complete procedure. Section 5.1 and Appendix D provide the models, datasets, evaluation protocol, and implementation settings. Appendix E reports the multi-seed analysis.

REFERENCES

Yongqi An, Xu Zhao, Tao Yu, Ming Tang, and Jinqiao Wang. Fluctuation-based adaptive structured pruning for large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 10865–10873, 2024. doi: 10.1609/aaai.v38i10.28960.

Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. SliceGPT: Compress large language models by deleting rows and columns. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=vXxardq6db.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7432–7439, 2020. doi: 10.1609/aaai.v34i05.6239.

Xiaodong Chen, Yuxuan Hu, Jing Zhang, Yanling Wang, Cuiping Li, and Hong Chen. Streamlining redundant layers to compress large language models. In *International Conference on Learning Representations*, volume 2025, pp. 30362–30383, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/4b00a351b41358965613c118e87dc28c-Paper-Conference.pdf.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018. URL https://arxiv.org/abs/1803.05457.

Elias Frantar and Dan Alistarh. SparseGPT: Massive language models can be accurately pruned in one-shot. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 10323–10337, 2023. URL https://proceedings.mlr.press/v202/frantar23a.html.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. URL https://arxiv.org/abs/2407.21783.

Neel Guha, Julian Nyarko, Daniel Ho, Christopher Ré, Adam Chilton, Aditya K, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel Rockmore, Diego Zambrano, Dmitry Talisman, Enam Hoque, Faiz Surani, Frank Fagan, Galit Sarfaty, Gregory Dickinson, Haggai Porat, Jason Hegland, Jessica Wu, Joe Nudell, Joel Niklaus, John Nay, Jonathan Choi, Kevin Tobia, Margaret Hagan, Megan Ma, Michael Livermore, Nikon Rasumov-Rahe, Nils Holzenberger, Noam Kolt, Peter Henderson, Sean Rehaag, Sharad Goel, Shang Gao, Spencer Williams, Sunny Gandhi, Tom Zur, Varun Iyer, and Zehua Li. LegalBench: A collaboratively built benchmark for measuring legal reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 44123–44279, 2023.

Babak Hassibi and David G. Stork. Second order derivatives for network pruning: Optimal brain surgeon. In *Advances in Neural Information Processing Systems*, volume 5. Morgan-Kaufmann, 1992. URL https://proceedings.neurips.cc/paper_files/paper/1992/hash/303ed4c69846ab36c2904d3ba8573050-Abstract.html.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Bairu Hou, Qibin Chen, Jianyu Wang, Guoli Yin, Chong Wang, Nan Du, Ruoming Pang, Shiyu Chang, and Tao Lei. Instruction-following pruning for large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 23894–23909, 2025. URL https://proceedings.mlr.press/v267/hou25b.html.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Wei Huang, Anda Cheng, and Yinggui Wang. GradPruner: Gradient-guided layer pruning enabling efficient fine-tuning and inference for LLMs. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=bxzJorqyYM.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. PubMedQA: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 2567–2577, 2019. doi: 10.18653/v1/D19-1259.

Song-ha Jo, Youngrok Ko, Sang-goo Lee, and Jinseok Seol. Task-aware block pruning with output distribution signals for large language models. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 6089–6107, 2026. doi: 10.18653/v1/2026.findings-eacl.320.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023. doi: 10.1145/3600006.3613165.

Qi Le, Enmao Diao, Ziyan Wang, Xinran Wang, Jie Ding, Li Yang, and Ali Anwar. Probe pruning: Accelerating LLMs through dynamic pruning via model-probing. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=WOt1owGfuN.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. LLM-Pruner: On the structural pruning of large language models. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/44956951349095f74492a5471128a7e0-Abstract-Conference.html.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. WWW'18 open challenge: Financial opinion mining and question answering. In *Companion Proceedings of The Web Conference 2018*, pp. 1941–1942, 2018. doi: 10.1145/3184558.3192301.

Yulong Mao, Kaiyu Huang, Changhao Guan, Ganglin Bao, Fengran Mo, and Jinan Xu. DoRA: Enhancing parameter-efficient fine-tuning with dynamic rank distribution. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11662–11675, 2024. doi: 10.18653/v1/2024.acl-long.626.

Xin Men, Mingyu Xu, Qingyu Zhang, Qianhao Yuan, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. ShortGPT: Layers in large language models are more redundant than you expect. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 20192–20204, 2025. doi: 10.18653/v1/2025.findings-acl.1035.

Omar Naim, Krish Sharma, Niyar R. Barman, and Nicholas Asher. TELL-TALE: Task efficient LLMs with task aware layer elimination. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 22616–22638, 2026. doi: 10.18653/v1/2026.findings-acl.1136.

Manuel Nonnenmacher, Thomas Pfeil, Ingo Steinwart, and David Reeb. SOSP: Efficiently capturing global correlations by second-order structured pruning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=t5EmXZ3ZLR.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. MedMCQA: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Proceedings of the Conference on Health, Inference, and Learning*, volume 174 of *Proceedings of Machine Learning Research*, pp. 248–260. PMLR, 2022. URL https://proceedings.mlr.press/v174/pal22a.html.

Kangyu Qiao, Shaolei Zhang, and Yang Feng. IG-pruning: Input-guided block pruning for large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 10618–10629, 2025. doi: 10.18653/v1/2025.emnlp-main.537.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8732–8740, 2020. doi: 10.1609/aaai.v34i05.6399.

Jiwon Song, Kyungseok Oh, Taesu Kim, Hyungjun Kim, Yulhwa Kim, and Jae-Joon Kim. SLEB: Streamlining LLMs through redundancy verification and elimination of transformer blocks. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 46136–46155, 2024. URL https://proceedings.mlr.press/v235/song24f.html.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=uyTL5Bvosj.

Yuanhe Tian, Junjie Liu, Xican Yang, Haishan Ye, and Yan Song. Frustratingly easy task-aware pruning for large language models. *arXiv preprint arXiv:2510.22489*, 2025. URL https://arxiv.org/abs/2510.22489.

Ziyan Wang, Enmao Diao, Qi Le, Pu Wang, Minwoo Lee, Shu-ping Yeh, Evgeny Stupachenko, Hao Feng, and Li Yang. From local to global: Revisiting structured pruning paradigms for large language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 35720–35739, 2026. doi: 10.18653/v1/2026.acl-long.1653.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Liangwei Yang, Yuhui Xu, Juntao Tan, Doyen Sahoo, Silvio Savarese, Caiming Xiong, Huan Wang, and Shelby Heinecke. Entropy-based block pruning for efficient large language models. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=bzQvL797PS.

Yifei Yang, Zouying Cao, and Hai Zhao. LaCo: Large language model pruning via layer collapse. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 6401–6417, 2024. doi: 10.18653/v1/2024.findings-emnlp.372.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019. doi: 10.18653/v1/P19-1472.

Mingyang Zhang, Hao Chen, Chunhua Shen, Zhen Yang, Linlin Ou, Xinyi Yu, and Bohan Zhuang. LoRAPrune: Structured pruning meets low-rank parameter-efficient fine-tuning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3013–3026, 2024. doi: 10.18653/v1/2024.findings-acl.178.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=lq62uWRJjiY.

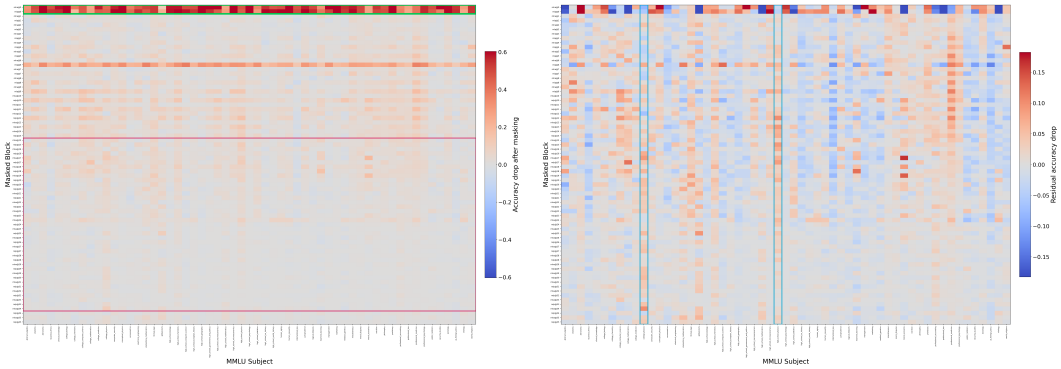
Longguang Zhong, Fanqi Wan, Ruijun Chen, Xiaojun Quan, and Liangzhi Li. BlockPruner: Fine-grained pruning for large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 5065–5080, 2025. doi: 10.18653/v1/2025.findings-acl.262.

Appendix

A Cross-Backbone Validation of the Motivating Observations

A.1 Block-Masking Responses on Qwen3-8B

To assess whether the motivating observations generalize across backbones, we repeat the MMLU block-wise masking study on Qwen3-8B. Figure 7 reports the resulting accuracy-drop and residual-response matrices.



(a) Accuracy drop (b) Residual drop

Figure 7: Qwen3-8B MMLU block-masking responses. (a) Accuracy drops from masking individual MHA or MLP blocks. (b) Residual drops after subtracting the generic component. Colored boxes mark the patterns discussed in the text.

The overall structure is consistent with that observed on Llama-3.1-8B, while exhibiting backbone-specific differences. In Figure 7(a), the green box covers only the two earliest blocks, compared with four on Llama-3.1-8B. This narrower concentration suggests that the shared sensitivity of Qwen3-8B is more localized among its earliest blocks, whereas the corresponding high-impact region extends across more early blocks in Llama-3.1-8B. Importantly, this difference concerns the spatial extent of the generic trend rather than its existence: both models contain early blocks whose removal consistently degrades performance across subjects. The purple box occupies a similar middle-to-late region in both models, where masking generally has a smaller effect. However, this region contains many negative drops on Llama-3.1-8B, whereas Qwen3-8B exhibits mostly small positive drops. The two backbones therefore agree that many later blocks are less sensitive under individual masking, but differ in whether masking them is neutral, mildly harmful, or occasionally beneficial; the present evidence does not isolate whether this difference arises from architecture or pretraining.

After subtracting the generic component, Figure 7(b) still reveals structured subject-specific variation. College physics and high-school physics, highlighted by the blue boxes, have a cosine similarity of 0.4041, while their mean similarity to all other subjects is -0.0605. Unlike the corresponding pattern on Llama-3.1-8B, electrical engineering does not join this pair: its cosine similarities with college physics and high-school physics are -0.0320 and -0.0728, respectively. These near-zero negative values indicate an absence of positive alignment rather than strong opposition. More broadly, this contrast cautions against interpreting residual similarity as a direct reproduction of coarse semantic categories such as “physics” or “STEM”: related subjects can exhibit different block-level deviations, and their groupings may change with the backbone. Thus, Qwen3-8B supports the same motivating premises of shared generic importance and non-arbitrary cross-task residual structure, while the particular salient blocks and subject groupings should be treated as model-dependent empirical patterns rather than universal task clusters.

A.2 Low-Rank Residual Spectrum on Qwen3-8B

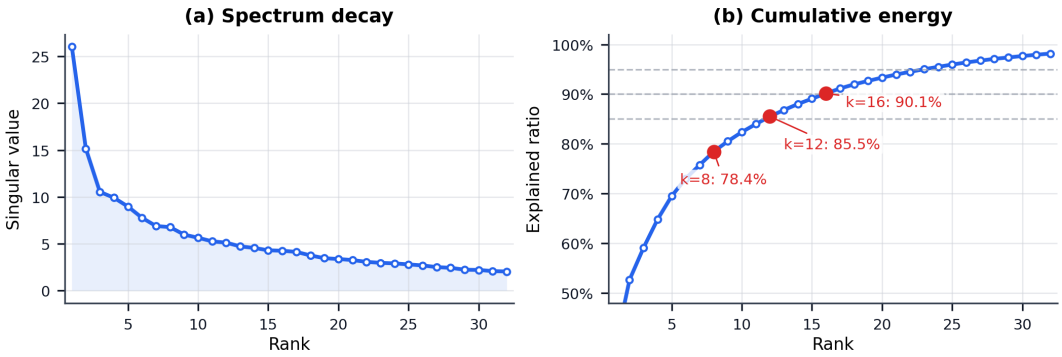


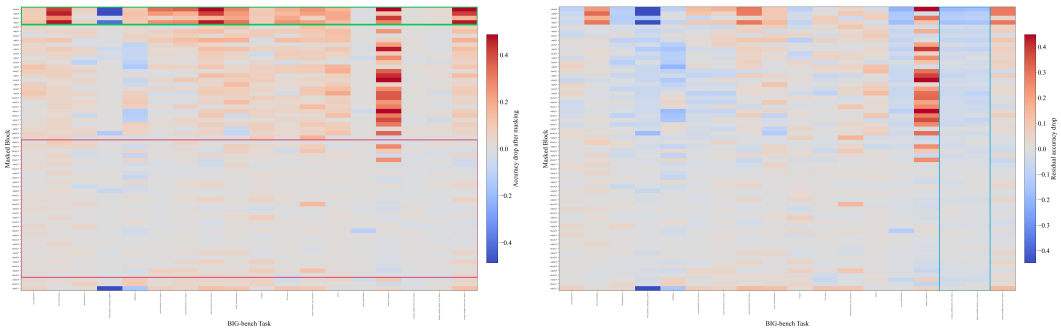
Figure 8: Singular-value spectrum of the Qwen3-8B MMLU residual matrix used for basis construction. (a) Spectrum decay. (b) Cumulative explained energy, with $k = 8$, 12, and 16 highlighted.

Figure 8 shows a concentrated spectrum. The leading direction explains 39.32% of the residual energy, while the top 8, 12, and 16 directions explain 78.44%, 85.50%, and 90.14%, respectively. Thus, Qwen3-8B distributes more residual variation beyond the leading components, but a compact subspace still captures most of its residual energy. Together with the block-masking patterns above, this result supports the two motivating observations across backbones while showing that their degree of concentration is model dependent.

B Cross-Dataset Validation of Basis Construction

B.1 Block-Masking Responses and Residual Spectrum from BIG-bench Tasks

To examine whether the low-rank structure depends on using MMLU as the reference-task collection, we select 18 direct multiple-choice tasks from BIG-bench. Each example has a unique correct answer. We use Llama-3.1-8B and at most 500 examples per reference task. All other residual construction and normalization steps follow the main method.



(a) Accuracy drop (b) Residual drop

Figure 9: Block-masking responses over 18 selected BIG-bench tasks on Llama-3.1-8B. (a) Accuracy drops from masking each of the 64 MHA/MLP blocks. (b) Residual drops after subtracting the generic component. Colored boxes mark patterns discussed in the text.

Figure 9 shows a shared depth-wise pattern across the selected tasks. The first four blocks highlighted in green have a mean accuracy drop of 0.159, compared with 0.025 across the remaining blocks, while many middle-to-late blocks highlighted in purple have relatively small effects. After subtracting the generic component, structured task-dependent variation remains. The blue box highlights the five- and seven-object variants of tracking shuffled objects. Their raw residual vectors

have a within-family cosine similarity of 0.943, compared with a mean cosine similarity of -0.062 to the other 16 tasks. These patterns provide qualitative support for shared generic importance and structured residual variation beyond MMLU. Because the collection contains only 18 tasks, including several related benchmark variants, these local similarities should not be interpreted as universal semantic task groupings.

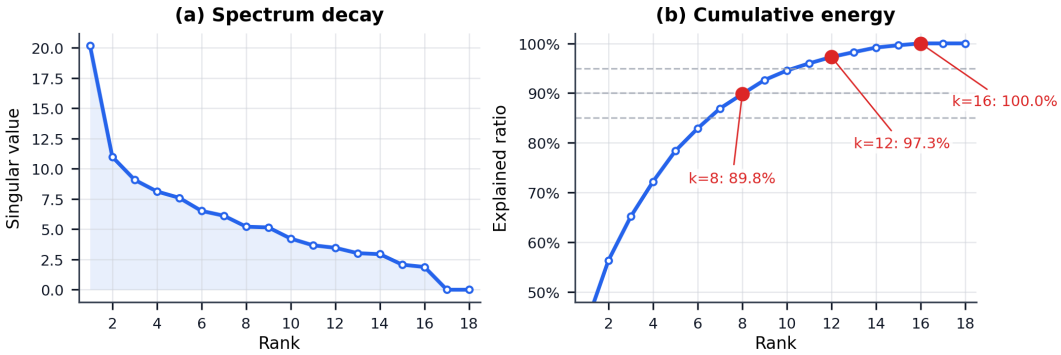


Figure 10: Singular-value spectrum of the residual matrix constructed from 18 selected BIG-bench reference tasks. (a) Spectrum decay. (b) Cumulative explained energy.

Complementing the response visualization, Figure 10 exhibits a concentrated spectrum: the first direction explains 43.46% of the residual energy, while the top 8 and 12 directions explain 89.81% and 97.27%, respectively. Compared with the MMLU spectrum in the main paper, its leading direction is less dominant (43.46% versus 51.75%), but its top 8 and 12 directions capture more energy (89.81% versus 85.15%, and 97.27% versus 90.37%). This indicates that the compact residual structure is not unique to MMLU, while the distribution of energy within that structure depends on the reference-task collection. The comparison should not be interpreted as showing intrinsically stronger low-rank structure for BIG-bench: its matrix has only 18 task columns, which imposes a substantially lower effective-rank ceiling than the 57-column MMLU matrix.

B.2 Effect of Reference-Task Composition on Downstream Pruning

To evaluate a mixed reference-task collection, we first align the MMLU and BIG-bench block-masking response matrices by their N block rows and concatenate their 57 and 18 task columns, producing a joint $N \times 75$ response matrix. We then recompute the block-wise generic component and task residuals over the combined columns, apply the same normalization, and perform SVD to learn a new basis. The joint basis is therefore refitted from the combined response matrix rather than formed by concatenating or averaging the two independently learned bases.

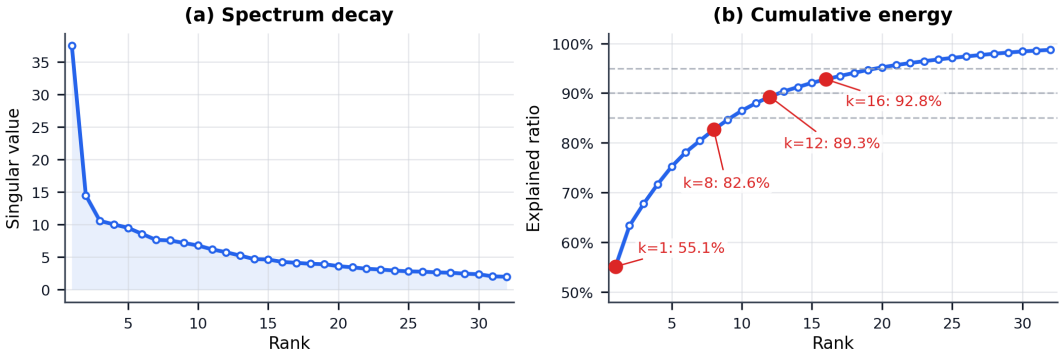


Figure 11: Singular-value spectrum of the joint residual matrix constructed from MMLU and 18 selected BIG-bench tasks. (a) Spectrum decay. (b) Cumulative explained energy.

As shown in Figure 11, the leading direction explains 55.10% of the residual energy, while the top 8, 12, and 16 directions explain 82.64%, 89.28%, and 92.78%, respectively. The joint matrix therefore remains strongly concentrated, but its shape is more nuanced than either source-specific spectrum. Its leading component is stronger than those of MMLU (51.75%) and the selected BIG-bench tasks (43.46%), whereas the energy outside the top eight dimensions is 17.36%, compared with 14.85% and 10.19%, respectively. Thus, combining the two collections produces both a more dominant first direction and a heavier spectral tail. One plausible interpretation is that the joint matrix contains a strong cross-collection direction together with secondary variations introduced by the more diverse reference tasks. This interpretation is descriptive rather than causal: singular values quantify concentration but do not identify the semantic content or dataset origin of individual directions.

We then compare residual bases constructed from MMLU, the 18 selected BIG-bench tasks, and their joint reference-task collection on the same nine non-MMLU downstream tasks. All three settings use $k = 8$, $q = 12$, a target parameter-pruning ratio of 25%, and supervised probe-based pruning without post-pruning fine-tuning, followed by zero-shot evaluation. Table 4 reports the resulting accuracies.

Basis source	**PubMed**	**MedMC**	**LICQ**	**FIQA**	**PIQA**	**Hella**	**Wino**	**ARC-E**	**ARC-C**	**Avg.**
MMLU	74.00	23.86	57.75	89.74	64.47	53.86	62.27	44.07	35.32	56.15
BIG-bench (18)	32.40	26.70	49.33	86.32	68.72	60.77	60.69	44.57	37.12	51.85
Joint (75)	61.80	26.32	42.08	89.32	68.77	57.74	68.19	51.39	36.86	55.83

Table 4: Zero-shot accuracy (%) of ReTaPER on Llama-3.1-8B under different reference-task compositions for residual-basis construction.

The joint reference-task collection achieves an average accuracy of 55.83%, improving over the 18-task BIG-bench basis by 3.98 points and remaining within 0.32 points of the MMLU basis. It improves six of the nine tasks over MMLU, including gains of 5.92 and 7.32 points on WinoGrande and ARC-Easy, respectively. These improvements are offset by drops of 12.20 points on PubMedQA and 15.67 points on LICQ. The result suggests that reference-task composition affects which downstream structural patterns are represented most effectively, rather than providing a uniform benefit from adding more tasks. It also aligns with the heavier tail of the joint spectrum: under the same $k = 8$ budget, a smaller fraction of its variation is retained. This offers a possible explanation for why the joint basis does not surpass MMLU on average, but does not establish that spectral truncation is the cause. In particular, the joint construction also recomputes generic importance and anchor selection, so the effects of basis rank, reference-task relevance, and the generic component are confounded. Taken together, these experiments show that ReTaPER is not methodologically tied to MMLU: a separately selected collection of BIG-bench tasks also yields a compact residual basis that transfers to non-reference downstream tasks, and the same pipeline remains applicable when the two reference collections are combined. MMLU is therefore one effective instantiation of the reference-task collection rather than a requirement of the framework. However, the accuracy differences also show that ReTaPER is not invariant to this choice; reference-task composition affects the learned subspace and its downstream transfer. We retain MMLU as the default because it provides the highest average accuracy with fewer reference tasks and lower offline construction cost than the joint collection.

C Complete ReTaPER Procedure

C.1 End-to-End Pseudocode

Algorithm 1 summarizes the complete workflow from offline residual-basis construction to task-specific pruning and dynamic LoRA adaptation. Here, $\mathrm{Acc}(f, \mathcal{D})$ denotes the accuracy of model f on dataset $\mathcal{D}$. $\mathrm{Bypass}(f, \mathcal{C})$ bypasses the blocks in $\mathcal{C}$ while leaving all other model components unchanged. $\mathrm{Params}(\cdot)$ denotes the number of parameters in a model or a set of blocks. Here, I_k is the $k \times k$ identity matrix, and $\mathbf{1}_d$ is the d-dimensional all-ones vector. The numerical tolerance τ prevents division by zero and determines when the selected anchor rows have exhausted the k-dimensional residual space.

Algorithm 1: Complete ReTaPER workflow

Require: Pretrained model f with blocks $\mathcal{B} = \{b_i\}_{i=1}^{N}$; reference-task datasets $\{\mathcal{D}_j\}_{j=1}^{J}$; target-task training data $\mathcal{D}_t$
Require: Basis rank $1 \le k \le \min(N, J)$ and anchor count $1 \le q \le N$
Require: Feasible pruning ratio $0 \le \alpha \le \mathrm{Params}(\mathcal{B})/\mathrm{Params}(f)$
Require: Ridge coefficient $\eta > 0$, rank-unit budget $B_t^{\mathrm{rank}} \in \mathbb{Z}_{\ge 0}$, concentration exponent $\gamma > 0$, rank-unit size $\kappa \in \mathbb{Z}_{>0}$, and tolerance $\tau > 0$
Ensure: Adapted pruned model f_t

Offline residual-basis construction

for $j = 1, \dots, J$ **do**
  $a_j^0 \leftarrow \mathrm{Acc}(f, \mathcal{D}_j)$
  for $i = 1, \dots, N$ **do**
    $a_{ij} \leftarrow \mathrm{Acc}(\mathrm{Bypass}(f, \{b_i\}), \mathcal{D}_j)$
    $m_{ij} \leftarrow a_j^0 - a_{ij}$
  end for
end for
for $i = 1, \dots, N$ **do**
  $g_i \leftarrow J^{-1} \sum_{j=1}^{J} m_{ij}$
  for $j = 1, \dots, J$ **do**
    $r_{ij} \leftarrow m_{ij} - g_i$
  end for
end for
$R \leftarrow [r_{ij}]_{N \times J}$
for $j = 1, \dots, J$ **do**
  $\mu_j \leftarrow N^{-1} \sum_{i=1}^{N} r_{ij}, \quad \sigma_j \leftarrow \max(\mathrm{Std}_i(r_{ij}), \tau)$
  $\widetilde{r}_{ij} \leftarrow (r_{ij} - \mu_j)/\sigma_j$ for all i
end for
$\widetilde{R} \leftarrow [\widetilde{r}_{ij}]_{N \times J}$
$[U_k, \Sigma_k, V_k^{\mathsf{T}}] \leftarrow \mathrm{TruncatedSVD}_k(\widetilde{R}); \quad \Phi \leftarrow U_k \Sigma_k$

Greedy anchor selection

Initialize the ordered anchor list $\mathcal{A} \leftarrow [\,]$ and $E \leftarrow \Phi$
▷ Apply the greedy coverage rule below, guided by the following objective for a small $\epsilon > 0$: $\max_{\substack{\mathcal{I} \subseteq \{1,\dots,N\} \\ |\mathcal{I}| = q}} \log\det(\Phi_{\mathcal{I}}^{\mathsf{T}} \Phi_{\mathcal{I}} + \epsilon I_k)$
while $|\mathcal{A}| < q$ **do**
  if $\max_{i \notin \mathcal{A}} \|E_{i,:}\|_2 > \tau$ **then**
    $a \leftarrow \arg\max_{i \notin \mathcal{A}} \|E_{i,:}\|_2$
    $v \leftarrow E_{a,:}/\|E_{a,:}\|_2; \quad E \leftarrow E - (Ev^{\mathsf{T}})v$
  else
    $a \leftarrow \arg\max_{i \notin \mathcal{A}} \|\Phi_{i,:}\|_2$
  end if
  Append a to $\mathcal{A}$
end while
$\Phi_{\mathcal{A}} \leftarrow \Phi[\mathcal{A}, :]$; cache $\{g_i\}_{i=1}^{N}$, Φ, and $\mathcal{A}$

Online task probing and pruning

Randomly sample $\mathcal{D}_t^p \subset \mathcal{D}_t$, and choose $\mathcal{D}_t^a \subseteq \mathcal{D}_t \setminus \mathcal{D}_t^p$
$a_t^0 \leftarrow \mathrm{Acc}(f, \mathcal{D}_t^p)$
for $\ell = 1, \dots, q$, with a_ℓ the ℓ-th index in $\mathcal{A}$ **do**
  $a_{a_\ell,t} \leftarrow \mathrm{Acc}(\mathrm{Bypass}(f, \{b_{a_\ell}\}), \mathcal{D}_t^p)$
  $s_{\ell,t} \leftarrow a_t^0 - a_{a_\ell,t} - g_{a_\ell}$
end for
$\widehat{\mu}_t \leftarrow q^{-1} \sum_{\ell=1}^{q} s_{\ell,t}, \quad \widehat{\sigma}_t \leftarrow \max(\mathrm{Std}_\ell(s_{\ell,t}), \tau)$
$\boldsymbol{\psi}_t \leftarrow ([s_{1,t}, \dots, s_{q,t}]^{\mathsf{T}} - \widehat{\mu}_t \mathbf{1}_q)/\widehat{\sigma}_t$
$\mathbf{z}_t \leftarrow (\Phi_{\mathcal{A}}^{\mathsf{T}} \Phi_{\mathcal{A}} + \eta I_k)^{-1} \Phi_{\mathcal{A}}^{\mathsf{T}} \boldsymbol{\psi}_t$
$\widehat{\mathbf{r}}_t \leftarrow \widehat{\mu}_t \mathbf{1}_N + \widehat{\sigma}_t \Phi \mathbf{z}_t$
$\widehat{d}_{i,t} \leftarrow g_i + \widehat{r}_{i,t}$ for all i; $\quad \pi_t \leftarrow \mathrm{argsort}_i\, \widehat{d}_{i,t}$
$\mathcal{P}_t \leftarrow \varnothing$
for i in the order π_t **do**
  if $\mathrm{Params}(\mathcal{P}_t)/\mathrm{Params}(f) \ge \alpha$ **then**
    break
  end if
  $\mathcal{P}_t \leftarrow \mathcal{P}_t \cup \{b_i\}$
end for

$\mathcal{R}_t \leftarrow \mathcal{B} \setminus \mathcal{P}_t$; $f_t \leftarrow \mathrm{Bypass}(f, \mathcal{P}_t)$
Residual-guided dynamic adaptation
$r_{\min} \leftarrow \min_{b_i \in \mathcal{R}_t} \widehat{r}_{i,t}$, $r_{\max} \leftarrow \max_{b_i \in \mathcal{R}_t} \widehat{r}_{i,t}$
if $r_{\max} - r_{\min} \le \tau$ **then**
  $\bar{r}_{i,t} \leftarrow 1$ for all $b_i \in \mathcal{R}_t$
else
  $\bar{r}_{i,t} \leftarrow (\widehat{r}_{i,t} - r_{\min})/(r_{\max} - r_{\min})$ for all $b_i \in \mathcal{R}_t$
end if
▷ Greedily solve $\max_{\{u_{i,t}\}} \sum_{b_i \in \mathcal{R}_t} (\bar{r}_{i,t})^\gamma \log(1 + u_{i,t})$, s.t. $\sum_{b_i \in \mathcal{R}_t} u_{i,t} = B_t^{\mathrm{rank}}$ and $u_{i,t} \in \mathbb{Z}_{\ge 0}$
$u_{i,t} \leftarrow 0$ for all $b_i \in \mathcal{R}_t$
for $h = 1, \ldots, B_t^{\mathrm{rank}}$ **do**
  $i^\star \leftarrow \arg\max_{b_i \in \mathcal{R}_t} (\bar{r}_{i,t})^\gamma \log((u_{i,t} + 2)/(u_{i,t} + 1))$
  $u_{i^\star,t} \leftarrow u_{i^\star,t} + 1$
end for
Assign LoRA rank $\mathrm{rank}_{i,t} \leftarrow \kappa u_{i,t}$ to each $b_i \in \mathcal{R}_t$
$f_t \leftarrow \mathrm{FineTuneLoRA}(f_t, \mathcal{D}_t^a)$, updating only adapters with $u_{i,t} > 0$
return f_t

D Detailed Experimental Settings

D.1 Models and Compute Environment

The main experiments use bf16 weights. Each run uses one NVIDIA L40 GPU with 48 GB of memory. The host system runs Ubuntu 20.04.4 and is equipped with an Intel Xeon Gold 6530 CPU and 251 GiB of memory. Transformers was version 4.47.0 for Llama-3.1-8B and 4.51.3 for Qwen3-8B; both environments used Python 3.9.20, PyTorch 2.5.1+cu121, Datasets 3.0.2, Evaluate 0.4.3, PEFT 0.13.2, TRL 0.13.0, NumPy 1.26.4, and SciPy 1.13.1. PyTorch used the CUDA 12.1 runtime with cuDNN 9.1, while the system CUDA toolkit was 11.8.

D.2 Data and Evaluation Protocol

The offline MMLU response matrix is constructed using at most 500 examples per subject. Two demonstrations from the corresponding dev subject precede each question. Prompts contain the subject description, question, four choices labeled A–D, and "Answer:"; candidates are scored by their continuation log-likelihood. Final downstream evaluation uses zero-shot prompting with the task templates summarized below; this designation does not apply to the supervised probe stage.

Task	Answer space / mapping
PubMedQA	yes, no, or maybe
MedMCQA	options A–D; `cop` is the gold index
LICQ	A: Yes; B: No
FIQA-SA	negative / neutral / positive by score sign
PIQA	two candidate solutions
HellaSwag	four candidate continuations
WinoGrande	two candidate fillers
ARC-E	answer-key-matched choices
ARC-C	answer-key-matched choices

Table 5: Answer spaces and label mappings for downstream evaluation.

All nine tasks are cast as multiple-choice problems. We score every candidate continuation and select the highest-scoring one, without sampling or free-form generation. The reported metric is the task-configured accuracy, using length-normalized candidate likelihood where specified by the evaluation harness. The nine-task average is an unweighted macro average over task accuracies. For ReTaPER, the probe and final-evaluation batch sizes follow the settings reported in the main paper.

D.3 ReTaPER Pruning and Numerical Details

The rank-k factorization uses `numpy.linalg.svd` with reduced matrices, and ridge coordinates are obtained by solving the regularized normal equations in float64. The default configuration is $k = 8$, $q = 12$, $\eta = 10^{-3}$, and $\alpha = 0.25$. Anchor selection first maximizes orthogonal coverage of the learned basis and then uses the largest remaining row norms, as described in the main paper and Algorithm 1.

D.4 LoRA Training Configuration

For an adapted MHA block, LoRA is inserted into `q_proj`, `k_proj`, `v_proj`, and `o_proj`; for an MLP block, it is inserted into `gate_proj`, `up_proj`, and `down_proj`. Dynamic allocation uses rank-unit size $\kappa = 4$, total unit budget $B_t^{\text{rank}} = |\mathcal{R}_t|$, and exponent $\gamma = 0.4$. Adapters are trained in bf16 for one epoch with at most 2500 task examples, maximum sequence length 1024, per-device batch size 8, and two gradient accumulation steps. We use AdamW with learning rate 2×10^{-4}, zero weight decay, a linear schedule with warmup ratio 0.03, LoRA scaling parameter 16, dropout 0.05, and no LoRA bias. Gradient checkpointing is enabled, sequence packing is disabled, and each supervised completion is terminated with the tokenizer EOS token.

E Multi-Seed Robustness and Statistical Uncertainty

To assess whether the main comparisons are stable to stochastic variation, the complete task-specific pipeline is repeated with five random seeds $\{42, 123, 12345, 123456, 100\}$. Where applicable, the seed controls probe-set sampling, adaptation-set sampling, training-data order, and LoRA initialization; final evaluation data remain fixed. Methods are paired by seed. For each backbone and setting, we report the mean and standard deviation of the nine-task macro average, together with ReTaPER's paired improvement over the strongest baseline and its 95% confidence interval. The interval is computed from the five seed-wise paired differences using a two-sided t-interval.

Backbone	Setting	Method	Avg. Acc.	Δ vs. Best	Paired 95% CI
Llama-3.1-8B	w/o FT	IG-Pruning	53.48 ± 0.63	–	–
		ReTaPER	56.15 ± 0.57	$+2.67$	$[2.16, 3.18]$
	w/ FT	LLM-Streamline	60.38 ± 0.59	–	–
		GradPruner	60.70 ± 0.55	–	–
		ReTaPER	66.97 ± 0.64	$+6.27$	$[5.08, 7.46]$
Qwen3-8B	w/o FT	IG-Pruning	50.75 ± 0.59	–	–
		ReTaPER	54.60 ± 0.53	$+3.85$	$[2.81, 4.89]$
	w/ FT	LLM-Streamline	58.88 ± 0.55	–	–
		GradPruner	58.57 ± 0.58	–	–
		ReTaPER	64.53 ± 0.83	$+5.65$	$[4.11, 7.19]$

Table 6: Multi-seed robustness of the principal comparisons in Table 1 of the main paper. Avg. Acc. is reported as mean ± standard deviation over five independent runs. Δ and its confidence interval compare ReTaPER with the strongest baseline under the same backbone and setting.

The multi-seed means preserve the average-accuracy ranking reported in the main-paper Table 1. Across the four backbone–setting combinations, ReTaPER improves over the strongest baseline by 2.67–6.27 points, and all paired 95% confidence intervals remain above zero. The run-to-run standard deviations are below one point for every method. These results support the stability of the comparisons.

F Residual-Subspace Structure and Validation

F.1 Geometry of the Leading Residual Directions

Figure 12 projects the fitted residual basis onto its first two directions, which jointly explain 62.5% of the residual energy. In panel (a), each point is a row of $\Phi = U_k \Sigma_k$ and therefore represents one masked block; panel (b) shows the corresponding MMLU coordinates encoded by V_k in $\widetilde{R} \approx \Phi V_k^{\mathsf{T}}$. The panels use separate numerical scales because block and task coordinates play different roles in this factorization.

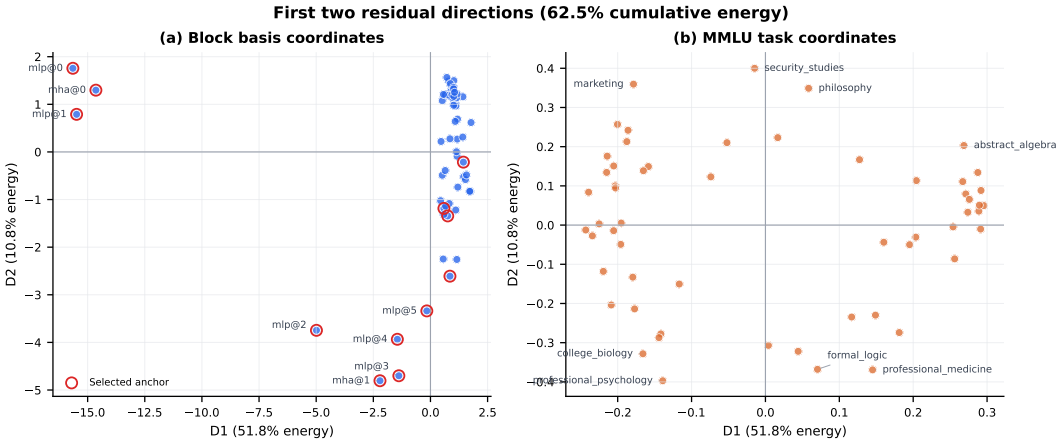


Figure 12: Geometry of the first two residual directions. (a) Block coordinates given by the first two columns of $\Phi = U_k \Sigma_k$; red outlines indicate the selected anchors. (b) MMLU subject coordinates in the same two directions. D_1 and D_2 explain 51.8% and 10.8% of the residual energy, respectively.

Several early MLP and MHA blocks lie far from the main block cluster along D_1, and most of them are selected as anchors. This provides a geometric interpretation of the coverage criterion: the selected rows include high-leverage locations that expose prominent residual directions, while also covering variation beyond the two dimensions shown here. On the task side, D_1 places many mathematical and physical subjects toward its positive extreme, whereas several biological and history subjects appear on the negative side. D_2 instead places security studies, philosophy, and marketing near its positive extreme, with several medical and biological subjects near its negative extreme. These associations offer a qualitative description of the fitted coordinates, not fixed semantic labels for the dimensions. SVD signs are arbitrary, and the omitted directions still account for 37.5% of the energy.

F.2 Task Coordinates in the Residual Subspace

Figure 13 provides a qualitative view of how reference and downstream tasks are represented in the rank-8 residual basis. The MMLU coordinates are obtained during basis construction, whereas each downstream coordinate is reconstructed from its anchor observations. To compare directional profiles independently of coordinate magnitude, each task vector is normalized to unit L_2 norm.

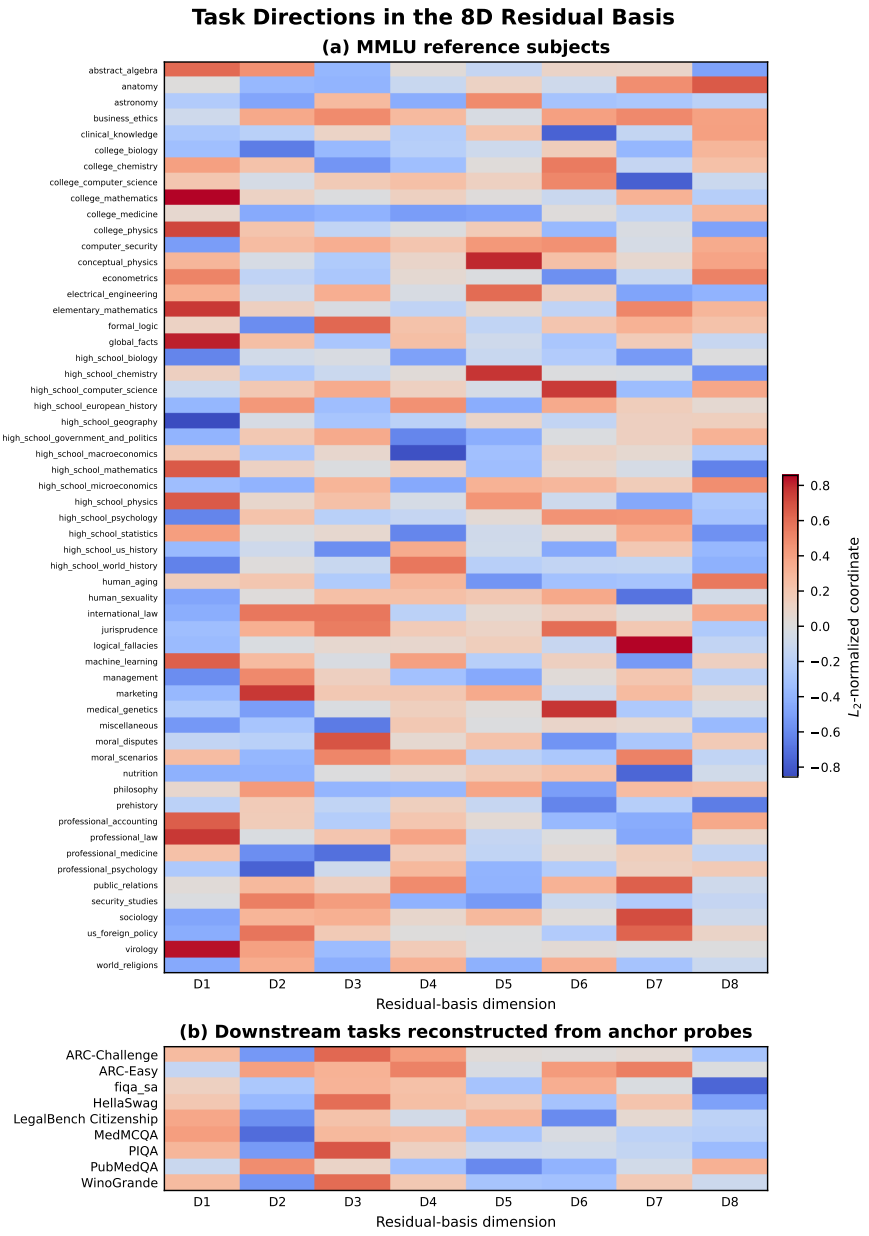


Figure 13: Task-coordinate directions in the rank-8 residual basis. (a) Coordinates of the 57 MMLU reference subjects obtained during SVD basis construction. (b) Coordinates of the nine downstream tasks reconstructed from anchor probes. Each row is L_2-normalized; color indicates the sign and relative contribution of each basis dimension.

The coordinate profiles vary substantially across both panels, showing that the task residuals are represented by different combinations of shared basis directions rather than by a single dominant template. Consistent with the two-dimensional view above, the reference subjects exhibit broad contrasts along D_1 and D_2, while D_3 further distinguishes several moral and logic-oriented subjects from medical subjects. Among the downstream tasks, D_3 contributes positively to most profiles, whereas PubMedQA is distinguished by a positive D_2 and negative D_5 contribution. These patterns provide possible qualitative associations for the leading directions, but they should not be interpreted as assigning fixed semantic meanings to individual dimensions: SVD signs are arbitrary, nearby directions may rotate under data perturbations, and the plotted basis describes one fitted model

rather than universal task categories. Quantitative reconstruction quality is evaluated separately in the main paper.

F.3 Null-Matrix Validation of Spectral Concentration

We test whether the concentrated residual spectrum can be explained by task marginals or heterogeneous variances alone. Starting from the 64×57 MMLU SVD input underlying Figure 4 of the main paper, we construct two null models. The column-permutation null independently permutes block indices within each standardized subject column, preserving its empirical value distribution while disrupting cross-subject alignment at the same blocks. The variance-profile sign null retains entrywise magnitudes, assigns independent random signs, and reapplies column centering and scale normalization. We apply the same subsequent transformations as for the observed SVD input, generate 2,000 matrices from each null with random seed 2027, and perform SVD.

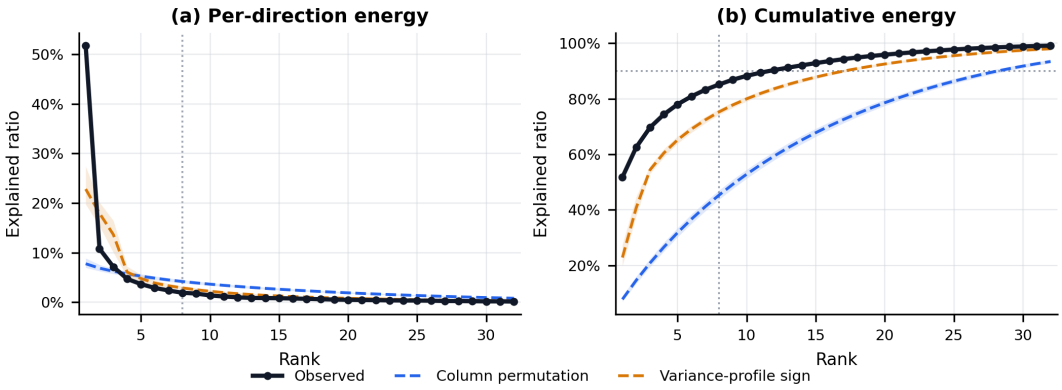


Figure 14: Observed MMLU residual spectrum and two randomized null distributions. Solid lines denote the observed matrix; dashed lines show null means, and shaded bands show the empirical 95% null intervals. The vertical dotted line marks $k = 8$, and the horizontal line in panel (b) marks 90% cumulative energy.

Metric	Observed	Column permutation	Variance-profile sign
E_1	51.75	7.75 [6.95, 8.87]	22.82 [19.75, 27.38]
E_8	85.15	45.32 [43.73, 47.02]	75.28 [74.21, 76.27]
E_{12}	90.37	59.41 [57.93, 60.98]	83.65 [82.87, 84.38]
E_{16}	93.59	70.23 [68.84, 71.63]	88.94 [88.36, 89.50]
Effective rank ↓	7.79	32.72 [31.83, 33.55]	15.13 [14.54, 15.66]
k_{90} ↓	12	28.66 [28, 29]	17.54 [17, 18]

Table 7: Observed spectral statistics and null means with empirical 95% null intervals. E_k is the cumulative energy retained by the top k singular directions, reported in percent.

As shown in Figure 14 and Table 7, the observed spectrum is substantially more concentrated than either null. Its top eight directions retain 85.15% of the energy, compared with null means of 45.32% and 75.28%. The observed effective rank is 7.79, versus 32.72 and 15.13, and it reaches 90% energy at rank 12 rather than approximately 29 or 18. None of the 2,000 matrices from either null attain the observed E_8; the add-one one-sided randomization test gives $p = 1/2001 < 5 \times 10^{-4}$.

These results indicate that the observed concentration is not reproduced by preserving either the per-task residual distributions or the standardized-entry magnitude profile under these randomizations. Instead, it reflects coherent block-level alignment across tasks. This test establishes separation from the specified null models, but does not by itself assess stability to finite-sample accuracy noise or changes in the reference-task subset.

G ACCURACY–EFFICIENCY TRADE-OFF OF SPARSE PROBING

G.1 COMPARISON WITH COMPLETE TASK PROBING

We compare ReTaPER with complete block-wise task scanning to quantify how residual approximation affects final pruning accuracy. The experiment uses Llama-3.1-8B, the same nine downstream tasks and probe sets as the main experiments, a target parameter-pruning ratio of 25%, and no post-pruning fine-tuning. For each task, Full-Scan Direct evaluates the full model and all 64 individually masked variants, then ranks blocks directly by the resulting task-specific accuracy drops. This is an empirical full-scan reference rather than a strict oracle, because independent block masking does not account for interactions among jointly pruned blocks.

We introduce two intermediate variants to separate approximation sources. Full-Observation Rank-8 observes all 64 block responses but constrains the reconstructed residual to the MMLU-derived rank-8 subspace. Anchor-12 Oracle Statistics estimates the task coordinates from only the 12 selected anchor responses while using the full residual vector's mean and standard deviation for normalization and denormalization. This latter variant is diagnostic rather than deployable: obtaining its oracle statistics still requires the complete block scan. ReTaPER Anchor-12 uses only the 12 anchor responses and estimates the normalization statistics from those anchors, as in the main method.

Variant	Avg. Acc. (%)	Probe Evals./Task	Probe time (min)	Speedup
Full-Scan Direct	59.89	65	219.97	$1.00\times$
Full-Observation Rank-8	58.35	65	219.97	$1.00\times$
Anchor-12 Oracle Statistics	56.48	65	219.97	$1.00\times$
ReTaPER Anchor-12	56.15	13	43.90	$5.01\times$

Table 8: Average pruning accuracy and probing cost over nine downstream tasks. Probe evaluations count, per task, the model variants evaluated on the complete probe set: one full model plus the corresponding block-masked variants. Probe time is aggregated over all nine tasks.

As shown in Table 8, constraining the fully observed residual to the rank-8 subspace reduces average accuracy from 59.89% to 58.35%, a gap of 1.54 points. Restricting coordinate estimation to 12 anchors, while retaining full-vector normalization statistics, introduces a further 1.87-point gap. Replacing these oracle statistics with the anchor-estimated statistics used by ReTaPER costs only another 0.33 points in this setting. Thus, the observed end-to-end gap is associated primarily with low-rank compression and sparse anchor observation, while the normalization approximation has a comparatively limited empirical effect here. This sequential comparison is descriptive rather than a strict causal decomposition, since small score changes can alter the discrete pruning order.

Overall, ReTaPER is 3.74 points below the full-scan reference, but reduces probe evaluations from 65 to 13 and total probing time by approximately 80%, yielding a $5.01\times$ speedup. The result therefore makes explicit the accuracy–efficiency trade-off: sparse probing does not reproduce complete task scanning without loss, but preserves substantial task-specific pruning performance at one-fifth of the online probing cost.

G.2 SENSITIVITY TO PROBE-SET SIZE

We further vary the number of examples used to evaluate each anchor while holding the residual basis, $k = 8$, $q = 12$, and the 25% pruning ratio fixed. Figure 15 reports the downstream average accuracy, mean residual-reconstruction correlation, and total wall-clock probe time over the same nine tasks.

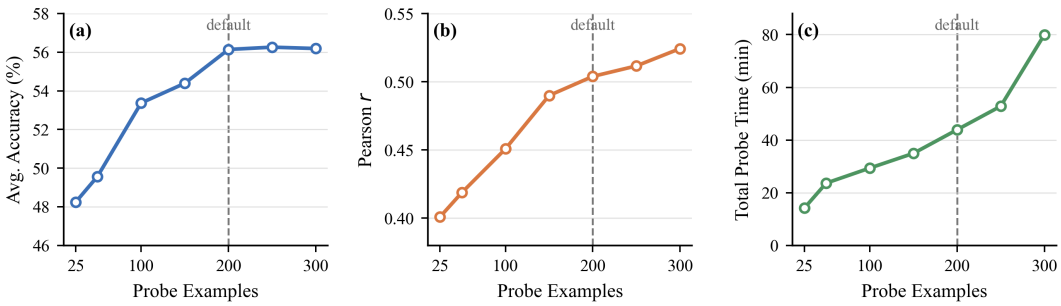


Figure 15: Effect of the per-task probe-set size. The three panels report nine-task average pruning accuracy, mean Pearson correlation between the reconstructed and observed residuals, and total probe time over all tasks. The dashed line marks the default $n_{\text{probe}} = 200$.

Increasing the probe set from 25 to 200 examples raises average accuracy from 48.24% to 56.15% and Pearson r from 0.4009 to 0.5039, showing that very small probe sets produce noisier task localization. Beyond 200 examples, correlation continues to improve, reaching 0.5243 at 300, whereas accuracy remains nearly unchanged: 56.26% at 250 and 56.19% at 300. This saturation suggests that finer residual reconstruction does not necessarily change the induced pruning order enough to improve final accuracy. We therefore retain 200 examples as the default accuracy–cost trade-off. Total probe time increases from 14.16 minutes at 25 examples to 79.98 minutes at 300, making the diminishing accuracy gains beyond 200 examples increasingly costly.

G.3 Anchor-Selection Ablation

We compare the anchor-selection heuristic used by ReTaPER with random selection, top-leverage selection, and greedy optimization of the regularized log-determinant objective. Top-leverage selection ranks candidate rows by their leverage scores in the rank-k residual subspace. All variants use the same basis, $k = 8$, $q = 12$, 200 probe examples per task, and a 25% pruning ratio. The random seeds affect only anchor selection.

Anchor Selection	**Avg. Acc. (%)** ↑	**Pearson** r ↑
Random (seed 0)	45.60	0.4417
Random (seed 1)	54.29	0.5013
Random (seed 2)	42.31	0.4169
Random (seed 3)	48.26	0.4881
Random (seed 4)	52.96	0.4913
Random (mean ± std.)	48.68 ± 5.00	0.4679 ± 0.0366
Top leverage	55.88	0.4993
Greedy log-det	56.13	**0.5042**
Orthogonal coverage (ReTaPER)	**56.15**	0.5039

Table 9: Ablation of anchor-selection strategies. Accuracy and correlation are averaged over the nine downstream tasks. Standard deviations for random selection are computed over five anchor sets.

Random anchors yield 48.68% accuracy on average with a 5.00-point standard deviation, showing that reconstruction is sensitive to whether the selected rows cover informative residual directions. Top-leverage selection is substantially stronger but remains 0.27 points below orthogonal coverage. Greedy log-det and the ReTaPER heuristic are effectively matched: their average accuracies differ by only 0.02 points, while their Pearson correlations differ by 0.0003. Thus, explicitly promoting directional coverage is important, and the simpler orthogonal-coverage heuristic closely approximates the empirical performance of direct log-det optimization in this setting. Since every strategy selects the same number of anchors, their online probe counts are identical; only the offline anchor-selection computation differs.

H Discussion

H.1 Scope of Independent Block Importance

ReTaPER estimates task-specific importance from independently masked block responses and uses the resulting scores to construct a fixed pruning order. It therefore does not explicitly model non-additive interactions among blocks removed together. This is a deliberate efficiency approximation shared by many structured pruning methods that use marginal block- or layer-level scores to reduce importance-estimation cost (Men et al., 2025; Yang et al., 2026).

Here, *interaction-aware importance* refers to an importance model in which a block's estimated contribution depends on which other blocks have already been or will be removed, rather than remaining a fixed score measured with that block masked in isolation. Such interactions can be addressed through conditional rescoring after each removal, joint mask optimization, or second-order structured pruning. For example, off-diagonal Hessian terms can approximate correlations among parameters or structures (Hassibi & Stork, 1992; Nonnenmacher et al., 2022). They do not, however, recover exact multi-block effects, and their computational cost depends on the curvature approximation. A more direct alternative is to evaluate every pair of blocks under joint masking, but this would require $O(N^2)$ masked-model evaluations, compared with $O(N)$ for an independent block-wise sweep. Iterative rescoring over K removals would instead require $O(KN)$ evaluations.

Our objective is not to establish independent scoring as universally preferable, but to investigate whether marginal task-specific block responses exhibit reusable low-rank structure and can be reconstructed from sparse anchor probes. Modeling conditional interactions would change the reconstruction target from an N-dimensional response vector to a pruning-set-dependent function and substantially increase online probing cost. We therefore retain independent block importance as an efficiency-oriented proxy. Accordingly, the Full-Scan Direct result in Section G.1 is a full-observation reference for marginal block scoring rather than a strict joint-pruning oracle. Designing interaction-aware residual representations would substantially exceed the scope of this work, but constitutes an important direction for future research.

H.2 Scope of Cost-Agnostic Block Ranking

ReTaPER adopts an importance-first policy: MHA and MLP blocks are ranked by their reconstructed task-specific importance, and pruning stops when the cumulative parameter ratio reaches the target. Because the two block types can have different parameter counts, this rule is an approximate selection policy rather than an exact solution to a parameter-constrained subset-selection problem. Similar importance-first ranking is common in prior block-pruning methods, which likewise separate importance estimation from resource-aware selection (Zhong et al., 2025; Yang et al., 2026).

The appropriate block cost is also deployment-dependent. MLP blocks typically contain more parameters, whereas removing MHA blocks can reduce KV-cache use; attention FLOPs vary with sequence length, and GQA assigns different dimensions to query/output and key/value projections. Actual latency further depends on kernels, batch size, sequence length, and hardware utilization. Indeed, BlockPruner reports greater inference speedups when more MHA blocks are removed at comparable parameter-pruning ratios, attributing the gain to reduced KV-cache use (Zhong et al., 2025). Thus, simple parameter normalization would optimize only one cost proxy and need not reflect deployment efficiency. We retain the original importance ranking to isolate task-specific importance reconstruction under a standard block-pruning protocol. A parameter-, FLOP-, memory-, or latency-aware downstream selector can replace this approximate rule without changing the residual reconstruction framework.

H.3 Implications

The supplementary results clarify both the scope and broader implications of ReTaPER. Low-rank task-residual structure is observed across backbones and distinct reference-task collections, suggesting that it is not confined to a single model or dataset. The randomized null comparison shows that the observed spectral concentration is not reproduced by the specified marginal-preserving nulls, while the multi-seed evaluation supports the stability of the downstream comparisons. Complete-probing experiments and ablations further characterize the accuracy–efficiency trade-off: low-rank

projection and sparse anchor observation account for most of the gap to complete probing, whereas anchor-estimated normalization has only a limited effect. Informative anchor selection is substantially more reliable than random selection, and pruning accuracy saturates near the default probe-set size even as residual-reconstruction correlation continues to improve. Taken together, results across two backbones, two distinct reference-task collections, and nine downstream tasks provide a coherent body of empirical evidence for the structured low-rank nature of task residuals and their utility in task-aware pruning and recovery.

H.4 Limitations and Future Directions

The scope of the evidence is limited to the models and task families evaluated in this work. Although the supplementary analyses examine residual patterns on Qwen3-8B and selected BIG-bench tasks, the end-to-end pruning and recovery experiments cover two 8B backbones and nine selected multiple-choice tasks. These results provide evidence across the evaluated backbones and tasks, but do not establish that the same residual organization or performance gains extend to substantially larger models, generative workloads, or broader out-of-distribution task distributions. We therefore do not claim that ReTaPER's current results predict its behavior in those settings.

The learned residual basis and its downstream utility also depend on the composition of the reference tasks. A different reference-task collection may produce a different basis and alter how informative a given anchor set is for a new task. Sparse anchor probing offers an accuracy–cost trade-off rather than recovering all information available from complete block observation. In the Llama-3.1-8B, no-fine-tuning evaluation across nine tasks, ReTaPER Anchor-12 reduced block evaluations from 65 to 13, yielding a $5.01\times$ probing speedup, while mean accuracy was 3.74 percentage points below the full-scan reference (Section G.1). The reconstruction-correlation metric captures signal-level association, but does not directly quantify agreement between the resulting block-importance ranking and the ranking induced by complete observation. Accordingly, we interpret the reconstructed residual as a task-dependent proxy for the full-observation ranking. In addition, ReTaPER estimates responses to independently masked blocks. It does not explicitly model interactions among blocks or optimize the joint effect of removing a particular set of blocks.

The method also shifts a substantial portion of computation to offline preprocessing. In the reported nine-task setting, offline processing takes 202.60 minutes per backbone, while online pruning and recovery take 63.89 minutes in total. Thus, the lower online cost comes with a substantial upfront cost, and the practical benefit depends on reusing the same backbone across multiple downstream tasks. These wall-clock measurements reflect the evaluated implementation and setup. Finally, ReTaPER allocates LoRA capacity under a fixed sum-of-ranks budget. For ease of implementation and comparison, we use this as an approximate allocation budget: because the target projections have different dimensions, equal rank sums do not necessarily imply equal numbers of trainable parameters. Accordingly, the rank-allocation results compare allocations under a fixed rank-unit budget, rather than under a strict parameter-matched budget.

Future work should evaluate ReTaPER on generative tasks, larger backbones, and a broader range of task distributions; study how reference-task selection and anchor choice affect transfer; and develop methods that account for interactions among jointly pruned blocks. Further work could reduce offline preprocessing cost, examine subspace stability under finite-sample noise and different reference-task subsets, and compare rank-allocation strategies under explicitly matched trainable-parameter budgets.