DEEP LOW-RANK PROJECTOR FOR LOW-RANK KV CACHE COMPRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Key-value (KV) cache memory grows linearly with context length and becomes a major bottleneck in long-context language model inference. We propose **Deep Low-Rank Projector (DLRP)**, a framework for compressing the KV cache along the head dimension of pretrained attention models. DLRP first trains full-dimensional Deep Linear Projectors (DLPs) on a frozen backbone using a KL objective and a regularizer that encourages low-rank structure. We establish an upper-bound relation between this regularizer and the nuclear norm of the composed projection matrix, providing a tractable surrogate that avoids direct nuclear-norm computation during training. Layer-wise reduced dimensions are then selected from the learned singular-value spectra, and the trained projectors are reduced and fine-tuned for recovery using only a small amount of data. Finally, the resulting DLRPs are folded into the attention projections, leaving no additional projector modules in the decoding path. Across Qwen3 and Llama-3 models, DLRP preserves downstream performance better than the compared hidden-dimension KV-cache compression methods over a range of compression ratios, remains compatible with token-axis compression, and reduces long-context memory usage while increasing feasible batch size.

1 INTRODUCTION

Large Language Models (LLMs) rely on the Key-Value (KV) cache to avoid recomputing past hidden states during autoregressive decoding (Pope et al., 2023). While this substantially reduces computation, the KV cache grows linearly with sequence length and becomes a major memory bottleneck for long-context inference (Shi et al., 2024). Existing KV-cache compression methods exploit redundancy along several axes, including layers, attention heads, tokens, and the head dimension (Brandon et al., 2024; Sun et al., 2024; Wu & Tu, 2024; Zuhri et al., 2024; Liao & Vargas, 2024; Goldstein et al., 2024; Shazeer, 2019; Ainslie et al., 2023; Devoto et al., 2024; Wang et al., 2024; Xiao et al., 2023; Li et al., 2024; Cai et al., 2024; Feng et al., 2024; Park et al., 2025). Among them, head-dimension compression reduces the size of each cached key/value vector and can be combined naturally with token-axis compression for additional memory savings (Saxena et al., 2024; Lin et al., 2024; Zhang et al., 2024; Chang et al., 2025; Yan et al., 2025; Liu et al., 2024).

While KV compression along the head dimension is promising, several practical challenges remain. One line of work employs non-foldable linear projections that persist throughout the decoding path, introducing additional computation during inference (Lin et al., 2024). Another line of work (Saxena et al., 2024; Zhang et al., 2024; Chang et al., 2025; Yan et al., 2025) relies on local proxy criteria, such as layer-wise sensitivity, reconstruction error, or head-wise statistics, to determine the reduced dimension. These criteria are defined at the level of individual layers or heads, whereas the resulting compressed model is ultimately evaluated by its end-to-end behavior. Finally, models designed with reduced head dimensions (Liu et al., 2024) typically require substantial additional training, making them less straightforward to apply to existing pretrained models.

We evaluate DLRP on Qwen3 and Llama-3 models across short-context benchmarks, LongBench, and RULER, together with representative hidden-dimension KV-cache compression baselines. We also evaluate DLRP in combination with token-axis compression methods. Across the evaluated compression ratios, DLRP maintains higher downstream performance than the compared hidden-dimension baselines, while adding only limited performance degradation when combined with

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.

token-axis compression. We further evaluate long-context serving using a custom variable-rank Triton attention kernel, where the reduced head dimensions lead to lower peak memory and larger feasible batch sizes.

Our contributions are summarized as follows:

- We propose **DLRP**, a KV-cache compression framework that learns layer-wise reduced head dimensions from full-dimensional DLPs, fine-tunes the reduced projectors for recovery, and folds them into the pretrained attention layers for inference.
- We introduce a regularizer that encourages low-rank structure in DLPs and theoretically establish an upper-bound relation to the nuclear norm of the composed projection matrix. This provides a tractable alternative to direct nuclear-norm computation during training.
- We develop a two-stage procedure that first learns full-dimensional DLPs on a frozen backbone and then selects layer-wise reduced dimensions from their learned singular-value spectra, followed by reduction and recovery fine-tuning.
- We evaluate DLRP across multiple model families, short- and long-context benchmarks, hidden-dimension compression baselines, and hybrid token-axis compression settings.

2 Related Work

2.1 Deep Linear Networks

Deep Linear Networks (DLNs) have been widely studied as analytical models for understanding optimization and generalization in deep learning. Despite representing only linear input–output mappings, their factorized parameterization gives rise to non-trivial optimization dynamics and implicit biases (Saxe et al., 2013; Kawaguchi, 2016; Hardt & Ma, 2016; Laurent & Brecht, 2018). DLNs have also provided theoretical insights into over-parameterized linear models (Ziyin et al., 2022; Hastie et al., 2022) and modern learning phenomena such as posterior collapse (Lucas et al., 2019; Wang & Ziyin, 2022) and neural collapse (Tian, 2022). In contrast to prior work that primarily uses DLNs as analytical tools, we employ their factorized parameterization as trainable projectors for post-training KV-cache compression. The resulting DLPs are optimized with a behavior-preserving KL objective and a factor-wise regularizer, and their learned spectra are subsequently used to determine the reduced dimensions of DLRPs.

2.2 Low-Rank KV Cache Compression

Low-rank, or hidden-dimension, KV-cache compression reduces the size of each cached key and value representation and is complementary to token-axis compression. MatryoshkaKV (Lin et al., 2024) reduces the head dimension through learned linear projections that remain in the decoding path, introducing additional projection operations at inference time. Other approaches, including Eigen-Attention, LoRC, Palu, and ReCalKV (Saxena et al., 2024; Zhang et al., 2024; Chang et al., 2025; Yan et al., 2025), determine reduced dimensions using criteria such as reconstruction error, sensitivity or importance estimates, and head-wise statistics under a target compression budget. Architectural designs such as MLA-style attention intrinsically use low-dimensional latent representations, but retrofitting such architectures to existing pretrained models generally requires substantial additional training (Liu et al., 2024).

DLRP instead learns full-dimensional, factorized projectors on a frozen pretrained model and selects layer-wise reduced dimensions from their learned singular-value spectra. The corresponding boundary factors (output-side for query/key/value and input-side for the output projection) are then reduced via SVD and fine-tuned for recovery. Finally, the resulting DLRPs are folded into the attention projections, leaving no additional projection modules in the decoding path.

3 BACKGROUND

3.1 KEY-VALUE (KV) CACHE IN TRANSFORMER

In an L-layer transformer, the hidden representation $x_t \in \mathbb{R}^{1\times d_{\text{embed}}}$ at decoding step t and layer l is projected as

$$q_l^t = x_t W_{Q_l}, \qquad k_l^t = x_t W_{K_l}, \qquad v_l^t = x_t W_{V_l}, \tag{1}$$

where $W_{Q_l}, W_{K_l}, W_{V_l} \in \mathbb{R}^{d_{\text{embed}}\times d_{\text{head}}}$.

During autoregressive decoding, the new key and value vectors are appended to the cache:

$$K_l^t = \text{Concat}(K_l^{t-1}, k_l^t), \qquad V_l^t = \text{Concat}(V_l^{t-1}, v_l^t) \in \mathbb{R}^{t\times d_{\text{head}}}, \tag{2}$$

with $K_l^{t-1}, V_l^{t-1} \in \mathbb{R}^{(t-1)\times d_{\text{head}}}$.

The attention output is

$$z_l^t = \text{Softmax}\left(\frac{q_l^t K_l^{t\top}}{\sqrt{d_{\text{head}}}}\right) V_l^t, \tag{3}$$

followed by the output projection $W_{O_l} \in \mathbb{R}^{d_{\text{head}}\times d_{\text{embed}}}$.

The KV cache avoids recomputing previous keys and values and has shape `[layers, heads, sequence length, head dimension]`. Reducing d_{head} therefore directly reduces KV-cache memory.

3.2 DEEP LINEAR NETWORKS (DLNS)

Deep Linear Networks (DLNs) are neural networks composed entirely of linear transformations without intermediate nonlinearities. Formally, an N-layer DLN f is defined as

$$f(x;\theta) = W_N W_{N-1} \cdots W_1 x, \qquad \theta = \{W_1, \ldots, W_N\}, \tag{4}$$

where $W_n \in \mathbb{R}^{d_n \times d_{n-1}}$ denotes the weight matrix of the n-th layer and $x \in \mathbb{R}^{d_0}$ is the input.

Although depth does not increase the expressive power of DLNs, since the composition remains a linear transformation, their factorized parameterization induces non-trivial optimization dynamics. In particular, DLNs exhibit non-convex optimization landscapes with multiple critical points and saddle points, and have been widely studied as simplified models for understanding optimization and implicit regularization in deep networks (Ge et al., 2015; Lee et al., 2016).

4 METHOD

We propose Deep Low-Rank Projector (DLRP), a post-training framework for KV-cache compression along the hidden dimension. DLRP trains full-dimensional Deep Linear Projectors (DLPs), reduces them using spectrum-based rank selection, and fine-tunes the resulting DLRPs for recovery before folding them into the attention projections for inference. The complete procedure is summarized in Appendix A.

4.1 DEEP LINEAR PROJECTOR

For the query projection $W_{Q_l} \in \mathbb{R}^{d_{\text{embed}}\times d_{\text{head}}}$ in layer l, we define a Deep Linear Projector (DLP) as

$$D_{Q_l} = D_{Q_l}^1 D_{Q_l}^2 \cdots D_{Q_l}^N \in \mathbb{R}^{d_{\text{head}}\times d_{\text{head}}}, \tag{5}$$

where $D_{Q_l}^1 \in \mathbb{R}^{d_{\text{head}}\times d_{\text{hidden}}}$, $D_{Q_l}^N \in \mathbb{R}^{d_{\text{hidden}}\times d_{\text{head}}}$, and $D_{Q_l}^n \in \mathbb{R}^{d_{\text{hidden}}\times d_{\text{hidden}}}$ for $n = 2, \ldots, N-1$. DLPs for the key, value, and output projections are defined analogously. To promote low-rank structure in these projectors, a common approach is to employ nuclear-norm regularization (Alvarez & Salzmann, 2017). However, computing the nuclear norm requires a Singular Value Decomposition (SVD) at each training iteration. With a computational complexity of $O(d^3)$, this can become costly for high-dimensional transformations. To address this issue, we introduce the following regularizer:

$$\mathcal{L}_{\text{Reg}} = \sum_{l=1}^{L}\sum_{n=1}^{N}\left(\|D_{Q_l}^n\|_F + \|D_{K_l}^n\|_F + \|D_{V_l}^n\|_F + \|D_{O_l}^n\|_F\right). \tag{6}$$

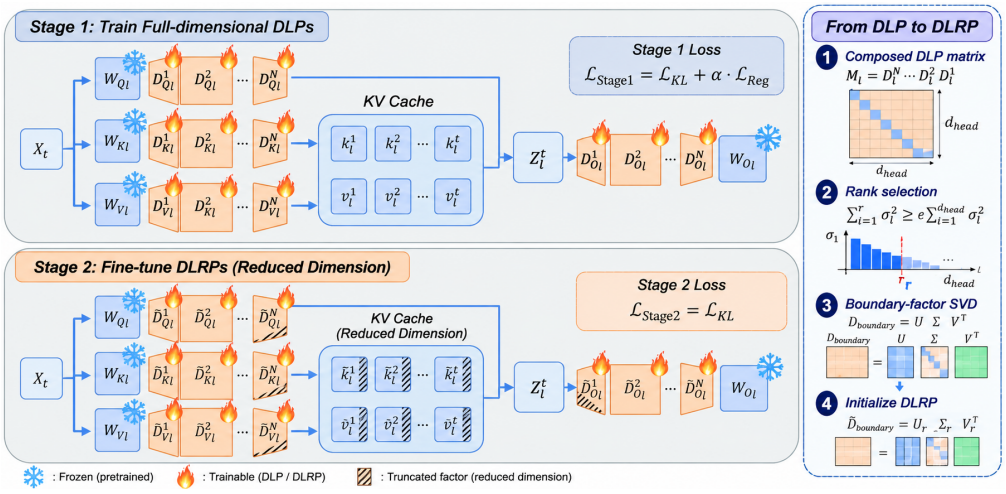

Figure 1: Overview of DLRP. Stage 1 trains full-dimensional DLPs with $\mathcal{L}_{\text{Stage1}} = \mathcal{L}_{\text{KL}} + \alpha\mathcal{L}_{\text{Reg}}$. Layer-wise ranks are selected from the spectra of the composed DLPs, and the corresponding boundary factors are reduced via SVD to construct DLRPs. Stage 2 fine-tunes the DLRPs with $\mathcal{L}_{\text{Stage2}} = \mathcal{L}_{\text{KL}}$ for recovery, followed by folding into the attention projections for inference.

The regularizer is related to the nuclear norm of the composed projector through the following upper bound:

Theorem 1. *Let $D^1 \cdots D^N \in \mathbb{R}^{d\times d}$ be a Deep Linear Projector. Then,*

$$\left\|D^1 \cdots D^N\right\|_* \leq \left(\frac{1}{N}\sum_{n=1}^{N}\|D^n\|_F\right)^N. \tag{7}$$

The proof is provided in Appendix B. Using tractable surrogate objectives or upper bounds for quantities that are expensive to optimize directly is a common strategy in optimization (Lange et al., 2000; Nguyen, 2017). Since N is fixed, minimizing the sum of factor-wise Frobenius norms also minimizes the corresponding upper bound up to an N-dependent monotonic transformation. We therefore use Eq. (6) as a tractable regularizer, reducing the corresponding computation from the $O(d^3)$ SVD required by direct nuclear-norm regularization to $O(d^2)$.

The Stage 1 objective is

$$\mathcal{L}_{\text{Stage1}} = \mathcal{L}_{\text{KL}} + \alpha\mathcal{L}_{\text{Reg}}, \tag{8}$$

where $\mathcal{L}_{\text{KL}}$ is the KL divergence between the original model and the model with DLPs, and α controls the regularization strength.

4.2 From DLP to DLRP

Rank Selection After Stage 1, we determine the reduced dimension of each attention projection from the singular-value spectrum of the corresponding trained DLP. For a composed DLP $D_l \in \mathbb{R}^{d_{\text{head}}\times d_{\text{head}}}$ with singular values $\sigma_1 \geq \cdots \geq \sigma_{d_{\text{head}}}$, we select the smallest rank r satisfying

$$\sum_{i=1}^{r}\sigma_i^2 \geq e\sum_{i=1}^{d_{\text{head}}}\sigma_i^2, \tag{9}$$

where $e \in (0, 1]$ is an energy threshold. Eq. (9) determines the target reduced dimension of each projector; the corresponding boundary factor is then reduced to this dimension as described below.

For the query and key projections, we first obtain individual ranks r^l_{query} and r^l_{key} and set

$$r^l_{\text{qk}} = \max\{r^l_{\text{query}}, r^l_{\text{key}}\}. \tag{10}$$

Similarly, for the value and output projections, we define

$$r^l_{\text{vo}} = \max\{r^l_{\text{value}}, r^l_{\text{output}}\}. \tag{11}$$

Query and key must share the same reduced dimension, and the same holds for value and output. We therefore take the maximum of the two individually selected ranks so that the selected dimension satisfies Eq. (9) for both DLPs in each pair.

DLP truncation. Given the selected dimensions, we decompose the corresponding boundary factor by SVD and reduce its factors to the target dimension while retaining the remaining trained DLP factors.

For the query projector, we decompose the final factor as

$$D^N_{Q_l} = U_{Q_l}\Sigma_{Q_l}V^\top_{Q_l}, \tag{12}$$

where, $U_{Q_l} \in \mathbb{R}^{d_{\text{hidden}}\times d_{\text{hidden}}}$, $\Sigma_{Q_l} \in \mathbb{R}^{d_{\text{hidden}}\times d_{\text{head}}}$, and $V^\top_{Q_l} \in \mathbb{R}^{d_{\text{head}}\times d_{\text{head}}}$. We reduce the output dimension of this factor as

$$\tilde{D}^N_{Q_l} = U_{Q_l}\Sigma_{Q_l}[:, : r^l_{\text{qk}}]V^\top_{Q_l}[: r^l_{\text{qk}}, : r^l_{\text{qk}}], \tag{13}$$

yielding $\tilde{D}^N_{Q_l} \in \mathbb{R}^{d_{\text{hidden}}\times r^l_{\text{qk}}}$. The preceding factors remain unchanged, i.e., $\tilde{D}^n_{Q_l} = D^n_{Q_l}$ for $n = 1, \ldots, N-1$. The resulting DLRP is

$$\tilde{D}_{Q_l} = \tilde{D}^1_{Q_l}\cdots\tilde{D}^N_{Q_l} \in \mathbb{R}^{d_{\text{head}}\times r^l_{\text{qk}}}. \tag{14}$$

The key and value projectors are reduced in the same manner using r^l_{qk} and r^l_{vo}, respectively.

For the output projector, the reduced dimension lies on the input side. We therefore decompose the first factor as

$$D^1_{O_l} = U_{O_l}\Sigma_{O_l}V^\top_{O_l}, \tag{15}$$

where $U_{O_l} \in \mathbb{R}^{d_{\text{head}}\times d_{\text{head}}}$, $\Sigma_{O_l} \in \mathbb{R}^{d_{\text{head}}\times d_{\text{hidden}}}$, and $V^\top_{O_l} \in \mathbb{R}^{d_{\text{hidden}}\times d_{\text{hidden}}}$. We reduce the input-side dimension as

$$\tilde{D}^1_{O_l} = U_{O_l}[: r^l_{\text{vo}}, : r^l_{\text{vo}}]\Sigma_{O_l}[: r^l_{\text{vo}}, :]V^\top_{O_l}, \tag{16}$$

yielding $\tilde{D}^1_{O_l} \in \mathbb{R}^{r^l_{\text{vo}}\times d_{\text{hidden}}}$. The remaining factors are unchanged, i.e., $\tilde{D}^n_{O_l} = D^n_{O_l}$ for $n = 2, \ldots, N$. This gives

$$\tilde{D}_{O_l} = \tilde{D}^1_{O_l}\cdots\tilde{D}^N_{O_l} \in \mathbb{R}^{r^l_{\text{vo}}\times d_{\text{head}}}. \tag{17}$$

The resulting DLRPs retain the trained factorized structure, with only the corresponding boundary factors reduced to the selected dimensions. Because rank selection is performed independently for each layer, these dimensions can vary across layers.

For RoPE-based models, Stage 1 is performed in the original d_{head}-dimensional space with RoPE unchanged. After dimensionality reduction, RoPE is applied in the reduced query/key dimension r^l_{qk}, and the scaled dot-product attention becomes

$$\tilde{Z}_l = \text{Softmax}\left(\frac{\tilde{Q}_l\tilde{K}_l^\top}{\sqrt{r^l_{\text{qk}}}}\right)\tilde{V}_l. \tag{18}$$

Further details on RoPE are provided in Appendix C.1.

4.3 Recovery Fine-Tuning and Deployment

Dimensionality reduction can introduce a performance drop. We therefore perform recovery fine-tuning while keeping the backbone frozen. During Stage 2, the DLRPs remain as separate factorized modules attached to the attention projections and are optimized using

$$\mathcal{L}_{\text{Stage2}} = \mathcal{L}_{\text{KL}}. \tag{19}$$

Stage 2 starts directly from the reduced Stage 1 factors and is used to recover the performance loss introduced by dimensionality reduction.

After recovery fine-tuning, the DLRPs are folded into the corresponding attention projections. For the query, key, and value projections,

$$\begin{aligned} W_{Q_l}\tilde{D}_{Q_l} &\to \tilde{W}_{Q_l} \in \mathbb{R}^{d_{\text{embed}} \times r^l_{\text{qk}}}, \\ W_{K_l}\tilde{D}_{K_l} &\to \tilde{W}_{K_l} \in \mathbb{R}^{d_{\text{embed}} \times r^l_{\text{qk}}}, \\ W_{V_l}\tilde{D}_{V_l} &\to \tilde{W}_{V_l} \in \mathbb{R}^{d_{\text{embed}} \times r^l_{\text{vo}}}. \end{aligned} \tag{20}$$

For the output projection, the DLRP is folded from the left:

$$\tilde{D}_{O_l} W_{O_l} \to \tilde{W}_{O_l} \in \mathbb{R}^{r^l_{\text{vo}} \times d_{\text{embed}}}. \tag{21}$$

After folding, no additional DLRP modules remain in the decoding path. The resulting model uses reduced-dimensional attention projections and stores smaller key and value representations in the KV cache.

5 Experiments

5.1 Experimental Setup

Models and Benchmarks. We evaluate DLRP on Qwen3-4B-Base, Qwen3-8B-Base (Yang et al., 2025), Llama-3.1-8B, and Llama-3.2-3B (Meta, 2024a;b). Short-context benchmarks include PIQA (Bisk et al., 2020), ARC-Challenge/Easy (Clark et al., 2018), HellaSwag (Zellers et al., 2019), Winogrande (Sakaguchi et al., 2021), and TQA (Kembhavi et al., 2017). For long-context evaluation, we use LongBench (Bai et al., 2024) and RULER Single-NIAH (Hsieh et al., 2024). Main results are reported on Qwen3-4B-Base and Qwen3-8B-Base; corresponding Llama results are provided in Appendices D.1 and D.2.

Baselines. We compare with hidden-dimension methods Eigen-Attention (Saxena et al., 2024), MatryoshkaKV (Lin et al., 2024), Palu (Chang et al., 2025), and ReCalKV (Yan et al., 2025). For token-axis compatibility, we use SnapKV (Li et al., 2024), StreamingLLM (Xiao et al., 2023), KeyDiff (Park et al., 2025), and PyramidKV (Cai et al., 2024) through NVIDIA KVPress (Jegou et al., 2024). For each baseline, we use the authors' released implementation when available; otherwise, we reproduce the method following the original paper.

Configuration and Training. Unless otherwise specified, we use $N = 2$, $d_{\text{hidden}} = 256$, and $\alpha = 0.05$, with Stage 1 DLP factors initialized to identity mappings. Layer-wise dimensions are selected using $e \in \{0.999, 0.990, 0.980, 0.970\}$ in Eq. (9); since the same e can yield different compression ratios across models, we report the resulting KV-cache compression ratio (CR) in the tables. The backbone remains frozen in both stages, using 100K and 20K samples drawn from SlimPajama-6B (Soboleva et al., 2023) for Stage 1 and Stage 2, respectively. Further training details are provided in Appendix C, and the effect of the Stage 2 recovery-set size is analyzed in Appendix E.6. For long-context and serving evaluations, we use a custom Triton attention kernel that supports the layer-wise reduced dimensions produced by DLRP. Implementation details are provided in Appendix C.2.

5.2 Main Results

5.2.1 Comparison with Hidden-Dimension KV-Cache Compression

Table 1 compares DLRP with hidden-dimension baselines at matched compression ratios, reporting the average over six short-context benchmarks. DLRP maintains the highest average performance among the evaluated hidden-dimension methods across all matched compression levels. On Qwen3-4B-Base, the score decreases only from 63.98 to 62.82 at 30.3% compression, whereas Eigen-Attention and Palu reach 58.00 and 55.40. On Qwen3-8B-Base, DLRP remains close to the backbone across the evaluated range, scoring 66.09 at 32.4% compression compared with the 66.36 backbone. The gap between DLRP and the other methods also remains visible as compression increases. Complete task-wise results across all model families and compression ratios are provided in Appendix D.1.

Table 1: Short-context average under matched hidden-dimension KV-cache compression ratios (CR).

Qwen3-4B-Base				*Qwen3-8B-Base*			
Method	**23.2%**	**27.3%**	**30.3%**	**Method**	**26.3%**	**30.1%**	**32.4%**
Backbone	63.98	63.98	63.98	Backbone	66.36	66.36	66.36
Eigen-Attention	61.83	59.23	58.00	Eigen-Attention	62.03	61.41	61.07
MatryoshkaKV	52.04	51.00	49.61	MatryoshkaKV	56.18	55.52	53.24
ReCalKV	51.30	47.72	44.55	ReCalKV	51.39	48.04	46.77
Palu	58.64	57.35	55.40	Palu	63.55	61.97	59.63
DLRP	**63.98**	**63.27**	**62.82**	**DLRP**	**66.77**	**66.30**	**66.09**

5.2.2 Long-Context Evaluation

LongBench. Figure 2 compares DLRP with hidden-dimension baselines at matched compression ratios on Qwen3-4B-Base and Qwen3-8B-Base. DLRP remains closer to the uncompressed backbone across the evaluated compression levels on both model scales. While all methods degrade as compression increases, the reduction is more gradual for DLRP.

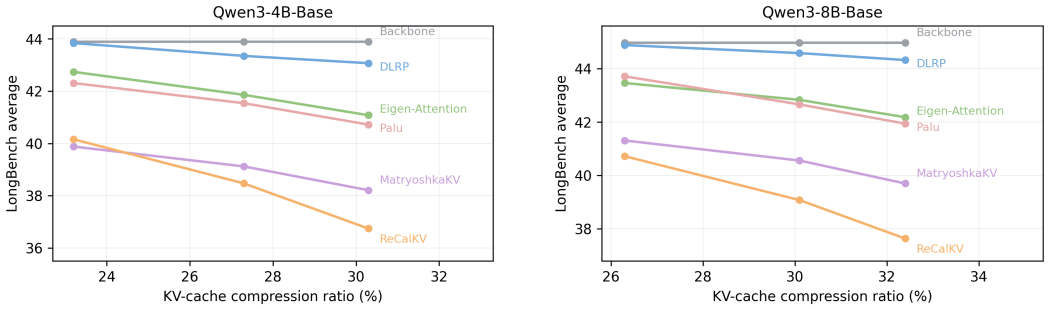

Figure 2: LongBench average under matched KV-cache compression ratios.

Table 2 reports category-wise results at representative compression ratios. The category-wise results show a similar trend. For Qwen3-4B-Base, the average changes from 43.89 to 43.07 at 30.3% compression, while Qwen3-8B-Base changes from 44.97 to 44.32 at 32.4%. The degradation is distributed across categories rather than being dominated by a single task type, with further decreases appearing at the higher compression settings. Complete category-wise results across compression ratios are provided in Appendix D.2.

Table 2: LongBench results of DLRP at representative compression ratios (CR).

Model	**CR**	**Single**	**Multi**	**Summ**	**FSL**	**Synth**	**Avg.**
Qwen3-4B	–	40.59	38.42	27.41	69.53	43.50	43.89
	30.3%	39.84	37.61	26.80	68.35	42.78	43.07
	32.9%	38.69	36.41	25.82	66.36	41.80	41.81
Qwen3-8B	–	36.53	42.75	27.48	69.78	48.33	44.97
	32.4%	35.98	41.95	27.02	68.88	47.78	44.32
	35.6%	35.08	40.69	26.28	67.44	46.97	43.29

RULER. We evaluate RULER Single-NIAH up to 32K context length. Table 3 compares DLRP with hidden-dimension baselines. MatryoshkaKV is omitted because its projection placement relative to RoPE does not use the same folded reduced-dimension execution path supported by our Triton kernel.

Table 3: RULER Single-NIAH accuracy (%) across context lengths.

	Qwen3-4B-Base					*Qwen3-8B-Base*				
Method	**4K**	**8K**	**16K**	**32K**	**Avg.**	**4K**	**8K**	**16K**	**32K**	**Avg.**
Backbone	99.6	99.4	99.0	98.4	99.1	100.0	100.0	100.0	100.0	100.0
Eigen-Attention	92.8	93.7	94.9	96.1	94.4	94.8	95.2	96.0	97.8	96.0
ReCalKV	92.1	93.2	94.5	95.6	93.9	94.2	95.0	96.2	97.2	95.6
Palu	93.6	94.7	95.8	96.9	95.3	95.1	95.8	96.6	98.0	96.4
DLRP	**95.0**	**95.8**	**96.9**	**98.0**	**96.4**	**96.3**	**96.7**	**98.5**	**99.3**	**97.5**

DLRP achieves the highest average accuracy among the compared compressed models on both Qwen3-4B-Base and Qwen3-8B-Base. The advantage is maintained across the evaluated context lengths, with DLRP reaching 98.0% and 99.3% at 32K. This indicates that reducing the hidden dimension with DLRP preserves long-range retrieval behavior well even as the context length increases.

5.2.3 Compatibility with Token-Axis Compression

Since DLRP and token-axis methods compress different KV-cache dimensions, they can be applied jointly. Table 4 reports representative combinations on LongBench using Qwen3-8B-Base. Adding DLRP results in only limited additional degradation across all four token-axis methods. The LongBench change is 0.62 for SnapKV, 0.56 for StreamingLLM, 0.43 for KeyDiff, and 0.26 for PyramidKV after adding 32.4% hidden-dimension compression. This consistent behavior across different token-axis strategies suggests that the two compression axes can be combined without a large additional performance loss. Complete results across token-axis methods, compression ratios, and LongBench categories are provided in Appendix D.3.

Table 4: DLRP combined with token-axis KV-cache compression on Qwen3-8B-Base.

Method	Token	DLRP CR	LongBench
SnapKV	60%	–	43.94
SnapKV + DLRP	60%	32.4%	43.32
StreamingLLM	60%	–	35.99
StreamingLLM + DLRP	60%	32.4%	35.43
KeyDiff	60%	–	34.78
KeyDiff + DLRP	60%	32.4%	34.35
PyramidKV	60%	–	43.38
PyramidKV + DLRP	60%	32.4%	43.12

5.3 Ablation Studies

We conduct ablations on the main design choices of DLRP. Unless otherwise specified, we report the average over six short-context benchmarks on Qwen3-4B-Base and Qwen3-8B-Base at approximately 30% KV-cache compression.

Table 5: Ablation of the two-stage procedure and layer-wise rank allocation. Two-stage variants use identical layer-wise dimensions; rank-allocation variants use the same total reduced-dimension budget.

Two-Stage Design			*Rank Allocation*		
Configuration	**4B**	**8B**	**Allocation**	**4B**	**8B**
Backbone	63.98	66.36	Uniform	61.96	65.61
Direct reduced DLRP	61.68	64.95	Random	61.43±0.27	65.18±0.24
Stage 1 → Reduction	61.94	65.42	**Spectrum-based**	**62.82**	**66.30**
Full DLRP	**62.82**	**66.30**			

Two-Stage Design. Table 5 compares the complete DLRP with a reduced DLRP trained directly from scratch using the same layer-wise dimensions. Direct training scores 61.68 and 64.95 on Qwen3-4B-Base and Qwen3-8B-Base, while full DLRP reaches 62.82 and 66.30. The Stage 1 → Reduction result further shows that Stage 2 recovers part of the performance lost during dimensionality reduction. Additional results across compression ratios and a stage-wise analysis of the recovery process are provided in Appendix E.2 and Appendix F.3.

Layer-Wise Rank Allocation. Table 5 compares spectrum-based allocation with uniform and random assignments under the same total reduced-dimension budget, with random allocation averaged over five samples. Spectrum-based allocation performs better on both model scales, indicating that the Stage 1 spectra provide a useful signal for distributing dimensions across layers.

Table 6: Effect of DLP depth N and hidden dimension d_{hidden} on Qwen3-4B-Base.

		d_{hidden}		
Benchmark	**Depth**	**128**	**256**	**512**
Short Task Avg.	$N = 1$	62.30	–	–
	$N = 2$	63.72	**63.98**	63.21
	$N = 3$	63.01	63.63	62.46
LongBench	$N = 1$	42.49	–	–
	$N = 2$	43.66	**43.87**	43.24
	$N = 3$	43.08	43.61	42.68

DLP Architecture. We study the DLP depth N and hidden dimension d_{hidden}. Table 6 shows that $N = 2$ and $d_{\text{hidden}} = 256$ perform best overall across short-context and LongBench evaluations; increasing depth or width provides no further gain. We therefore use this configuration in the main experiments. Configurations marked with "–" are omitted for the corresponding single-factor setting. Additional architecture results and alternative boundary-reduction strategies are provided in Appendices E.5 and E.4.

5.4 Analysis

Regularization and Learned Rank Structure. Figure 3 shows the layer-wise dimensions r^l_{qk} and r^l_{vo} under different values of α at the same energy threshold. As α increases, the selected dimensions generally decrease across both projection pairs. This empirical trend is consistent with the intended effect of the proposed regularizer: although Eq. (7) provides only an upper bound on the nuclear norm of the composed projection matrix, stronger regularization leads to spectra that require fewer singular components to satisfy the same energy threshold. The selected dimensions also vary across layers and between the query–key and value–output paths, resulting in non-uniform layer-wise compression. Additional sensitivity and spectral analyses are provided in Appendices E.1, F.1, and F.2.

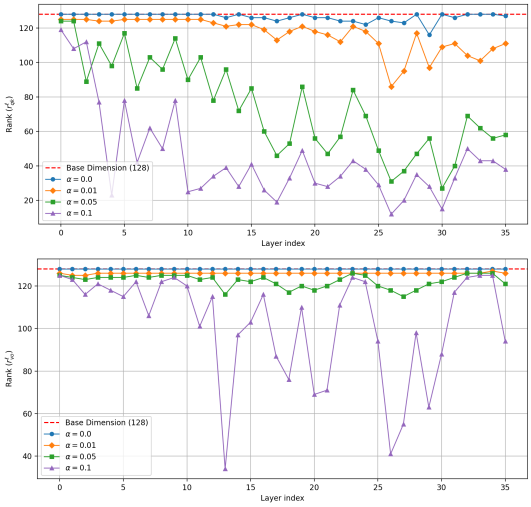

Figure 3: Layer-wise selected dimensions under different regularization strengths α at a fixed energy threshold.

Serving Efficiency. Because DLRP uses different reduced dimensions across layers, we implement a custom asymmetric variable-rank attention kernel in Triton for Qwen3-8B-Base. MatryoshkaKV is omitted because its projection placement relative to RoPE does not use the same folded reduced-dimension execution path supported by our kernel. Table 7 reports peak memory, maximum batch size, and decoding throughput at 16K and 64K context lengths. DLRP reduces peak memory and increases the maximum batch size from 14 to 23 at 16K and from 3 to 5 at 64K. Throughput is similar to the backbone at 16K, while the current Triton implementation is slower at 64K. After folding, no additional DLRP modules remain in the decoding path. Additional serving details are provided in Appendix F.5.

Table 7: Serving efficiency on Qwen3-8B-Base with a variable-rank Triton attention kernel.

Model	Context	Max BS	VRAM	Tok./s
Backbone	16K	14	69.65 GB	83.8
DLRP	16K	**23**	**65.11 GB**	**87.2**
Backbone	64K	3	61.89 GB	**14.4**
DLRP	64K	**5**	**58.64 GB**	11.0

6 Conclusion and Limitations

We proposed DLRP, a two-stage framework for reducing KV-cache memory through layer-wise head-dimension compression. DLRP learns full-dimensional projectors with a tractable low-rank regularizer, selects reduced dimensions from their spectra, and folds the recovered projectors into the attention layers for inference. Across short- and long-context benchmarks, DLRP preserves performance better than the compared hidden-dimension baselines and remains compatible with token-axis compression. A current limitation is that efficient deployment requires support for layer-wise variable dimensions, and our Triton implementation still shows lower throughput at very long context lengths.

AI USE STATEMENT

We used generative AI tools to assist with writing and debugging implementation code. We did not use generative AI tools to generate or analyze experimental results, create figures, or make scientific claims. All AI-assisted text and code were reviewed and verified by the authors, who take full responsibility for the final content of this work.

ETHICS STATEMENT

This work studies KV-cache compression for large language model inference and does not involve human subjects or the collection of private data. DLRP does not introduce new data sources or intended deployment settings beyond those of the underlying pretrained models, training corpus, and evaluation benchmarks.

REPRODUCIBILITY STATEMENT

Section 4 specifies the DLRP training procedure, regularization objective, spectrum-based rank selection, dimensionality reduction, recovery fine-tuning, and folding procedure. Section 5.1 describes the evaluated models, baselines, training data, compression settings, and benchmarks. Appendix A provides the complete training and deployment algorithms, while Appendix C provides additional training and implementation details, including the variable-rank Triton attention kernel. For each baseline, we use the authors' released implementation when available; otherwise, we reproduce the method following the original paper.

REFERENCES

Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.

Jose M Alvarez and Mathieu Salzmann. Compression-aware training of deep networks. *Advances in neural information processing systems*, 30, 2017.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. Longbench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 3119–3137, 2024.

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

VI Bityutskov. Bunyakovskii inequality, 2001.

William Brandon, Mayank Mishra, Aniruddha Nrusimha, Rameswar Panda, and Jonathan Ragan-Kelley. Reducing transformer key-value cache size with cross-layer attention. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Yucheng Li, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Junjie Hu, et al. Pyramidkv: Dynamic kv cache compression based on pyramidal information funneling. *arXiv preprint arXiv:2406.02069*, 2024.

Chi-Chih Chang, Wei-Cheng Lin, Chien-Yu Lin, Chong-Yan Chen, Yu-Fang Hu, Pei-Shuo Wang, Ning-Chi Huang, Luis Ceze, Mohamed S Abdelfattah, and Kai-Chiang Wu. Palu: Kv-cache compression with low-rank projection. In *The Thirteenth International Conference on Learning Representations*, 2025.

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, 2018. URL https://arxiv.org/abs/1803.05457.

Alessio Devoto, Yu Zhao, Simone Scardapane, and Pasquale Minervini. A simple and effective l_2 norm-based strategy for kv cache compression. *arXiv preprint arXiv:2406.11430*, 2024.

Yuan Feng, Junlin Lv, Yukun Cao, Xike Xie, and S Kevin Zhou. Ada-kv: Optimizing kv cache eviction by adaptive budget allocation for efficient llm inference. *arXiv preprint arXiv:2407.11550*, 2024.

Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on learning theory*, pp. 797–842. PMLR, 2015.

Daniel Goldstein, Fares Obeid, Eric Alcaide, Guangyu Song, and Eugene Cheah. Goldfinch: High performance rwkv/transformer hybrid with linear pre-fill and extreme kv-cache compression. *arXiv preprint arXiv:2407.12077*, 2024.

Moritz Hardt and Tengyu Ma. Identity matters in deep learning. *arXiv preprint arXiv:1611.04231*, 2016.

Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *Annals of statistics*, 50(2):949, 2022.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.

Simon Jegou, Maximilian Jeblick, and David Austin. kvpress, November 2024. URL `https://github.com/NVIDIA/kvpress`.

Kenji Kawaguchi. Deep learning without poor local minima. *Advances in neural information processing systems*, 29, 2016.

Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5376–5384, 2017. URL `https://api.semanticscholar.org/CorpusID:1310550`.

Kenneth Lange, David R Hunter, and Ilsoon Yang. Optimization transfer using surrogate objective functions. *Journal of computational and graphical statistics*, 9(1):1–20, 2000.

Thomas Laurent and James Brecht. Deep linear networks with arbitrary loss: All local minima are global. In *International conference on machine learning*, pp. 2902–2907. PMLR, 2018.

Jason D Lee, Max Simchowitz, Michael I Jordan, and Benjamin Recht. Gradient descent only converges to minimizers. In *Conference on learning theory*, pp. 1246–1257. PMLR, 2016.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. Snapkv: Llm knows what you are looking for before generation. *Advances in Neural Information Processing Systems*, 37:22947–22970, 2024.

Bingli Liao and Danilo Vasconcellos Vargas. Beyond kv caching: Shared attention for efficient llms. *arXiv preprint arXiv:2407.12866*, 2024.

Bokai Lin, Zihao Zeng, Zipeng Xiao, Siqi Kou, Tianqi Hou, Xiaofeng Gao, Hao Zhang, and Zhijie Deng. Matryoshkakv: Adaptive kv compression via trainable orthogonal projection. *arXiv preprint arXiv:2410.14731*, 2024.

Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024.

James Lucas, George Tucker, Roger B Grosse, and Mohammad Norouzi. Don't blame the elbo! a linear vae perspective on posterior collapse. *Advances in Neural Information Processing Systems*, 32, 2019.

Meta. Llama 3.1 model card. https://github.com/meta-llama/llama-models/blob/main/models/llama3_1/MODEL_CARD.md, 2024a.

Meta. Llama 3.2 model card. https://github.com/meta-llama/llama-models/blob/main/models/llama3_2/MODEL_CARD.md, 2024b.

Hien D Nguyen. An introduction to majorization-minimization algorithms for machine learning and statistical estimation. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, 7 (2):e1198, 2017.

Junyoung Park, Dalton Jones, Matthew J Morse, Raghavv Goel, Mingu Lee, and Chris Lott. Keydiff: Key similarity-based kv cache eviction for long-context llm inference in resource-constrained environments. *arXiv preprint arXiv:2504.15364*, 2025.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. *Proceedings of Machine Learning and Systems*, 5:606–624, 2023.

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

Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

Utkarsh Saxena, Gobinda Saha, Sakshi Choudhary, and Kaushik Roy. Eigen attention: Attention in low-rank space for kv cache compression. *arXiv preprint arXiv:2408.05646*, 2024.

Noam Shazeer. Fast transformer decoding: One write-head is all you need. *arXiv preprint arXiv:1911.02150*, 2019.

Luohe Shi, Hongyi Zhang, Yao Yao, Zuchao Li, and Hai Zhao. Keep the cost down: A review on methods to optimize llm's kv-cache consumption. *arXiv preprint arXiv:2407.18003*, 2024.

Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R Steeves, Joel Hestness, and Nolan Dey. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama. https://www.cerebras.net/blog/slimpajama-a-627b-token-cleaned-and-deduplicated-version-of-redpajama, 2023. URL https://huggingface.co/datasets/cerebras/SlimPajama-627B.

Yutao Sun, Li Dong, Yi Zhu, Shaohan Huang, Wenhui Wang, Shuming Ma, Quanlu Zhang, Jianyong Wang, and Furu Wei. You only cache once: Decoder-decoder architectures for language models. *Advances in Neural Information Processing Systems*, 37:7339–7361, 2024.

Yuandong Tian. Deep contrastive learning is provably (almost) principal component analysis. *arXiv preprint arXiv:2201.12680*, 3, 2022.

Zheng Wang, Boxiao Jin, Zhongzhi Yu, and Minjia Zhang. Model tells you where to merge: Adaptive kv cache merging for llms on long-context tasks. *arXiv preprint arXiv:2407.08454*, 2024.

Zihao Wang and Liu Ziyin. Posterior collapse of a linear latent variable model. *Advances in Neural Information Processing Systems*, 35:37537–37548, 2022.

Haoyi Wu and Kewei Tu. Layer-condensed kv cache for efficient inference of large language models. *arXiv preprint arXiv:2405.10637*, 2024.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.

Xianglong Yan, Zhiteng Li, Tianao Zhang, Haotong Qin, Linghe Kong, Yulun Zhang, and Xiaokang Yang. Recalkv: Low-rank kv cache compression via head reordering and offline calibration. *arXiv preprint arXiv:2505.24357*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

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

Rongzhi Zhang, Kuang Wang, Liyuan Liu, Shuohang Wang, Hao Cheng, Chao Zhang, and Yelong Shen. Lorc: Low-rank compression for llms kv cache with a progressive compression strategy. *arXiv preprint arXiv:2410.03111*, 2024.

Liu Ziyin, Botao Li, and Xiangming Meng. Exact solutions of a deep linear network. *Advances in Neural Information Processing Systems*, 35:24446–24458, 2022.

Zayd Muhammad Kawakibi Zuhri, Muhammad Farid Adilazuarda, Ayu Purwarianti, and Alham Fikri Aji. Mlkv: Multi-layer key-value heads for memory efficient transformer decoding. *arXiv preprint arXiv:2406.09297*, 2024.

A ALGORITHM

We summarize the complete DLRP pipeline in three steps: regularized DLP training, rank selection and dimensionality reduction, and recovery fine-tuning followed by deployment. The pretrained backbone remains frozen throughout both training stages.

Algorithm 1 Stage 1: Regularized DLP Training

Require: Frozen attention weights $\{W_{Q_l}, W_{K_l}, W_{V_l}, W_{O_l}\}_{l=1}^{L}$, training set $\mathcal{D}_1$, regularization strength α

Initialize full-dimensional factorized DLPs $\{D_{Q_l}, D_{K_l}, D_{V_l}, D_{O_l}\}_{l=1}^{L}$ as identity mappings
while not converged **do**
  Sample mini-batch $B \sim \mathcal{D}_1$
  Compute $\mathcal{L}_{\mathrm{KL}}$ between the backbone and DLP-attached model
  Compute $\mathcal{L}_{\mathrm{Reg}}$ using Eq. (6)
  Update only DLP factors using $\mathcal{L}_{\mathrm{KL}} + \alpha\mathcal{L}_{\mathrm{Reg}}$
end while
return Trained full-dimensional DLPs

Algorithm 2 Rank Selection and Construction of DLRP

Require: Trained DLPs and energy threshold e

for $l = 1, \ldots, L$ **do**
  Compute singular-value spectra of the composed D_{Q_l}, D_{K_l}, D_{V_l}, and D_{O_l}
  Select individual ranks using Eq. (9)
  Set $r^l_{\mathrm{qk}} = \max(r^l_{\mathrm{query}}, r^l_{\mathrm{key}})$
  Set $r^l_{\mathrm{vo}} = \max(r^l_{\mathrm{value}}, r^l_{\mathrm{output}})$
  Reduce $D^N_{Q_l}$, $D^N_{K_l}$, and $D^N_{V_l}$ by SVD to the selected output dimensions
  Reduce $D^1_{O_l}$ by SVD to the selected input dimension
  Retain all remaining trained Stage 1 factors unchanged
end for
return Reduced factorized DLRPs initialized directly from Stage 1

Algorithm 3 Stage 2 Recovery and Deployment

Require: Reduced DLRPs from Algorithm 2, recovery set $\mathcal{D}_2$

while not converged **do**
  Sample mini-batch $B \sim \mathcal{D}_2$
  Compute $\mathcal{L}_{\mathrm{Stage2}} = \mathcal{L}_{\mathrm{KL}}$
  Update only the reduced DLRP factors
end while
for $l = 1, \ldots, L$ **do**
  Fold $W_{Q_l}\tilde{D}_{Q_l}$, $W_{K_l}\tilde{D}_{K_l}$, and $W_{V_l}\tilde{D}_{V_l}$
  Fold $\tilde{D}_{O_l}W_{O_l}$
end for
return Reduced attention projections with no additional DLRP modules in the decoding path

B PROOF

We begin by stating the lemma, followed by the proof of the theorem in Section 4.1.

Lemma 1. *For any two matrices $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{n\times p}$, the following inequality holds:*

$$||AB||_F \leq ||A||_F \cdot ||B||_F \tag{22}$$

where $||\cdot||_F$ is the Frobenius norm.

Proof. Let $C = AB \in \mathbb{R}^{m\times p}$. By definition,

$$||C||_F^2 = \sum_{i=1}^{m}\sum_{j=1}^{p} c_{ij}^2 = \sum_{i=1}^{m}\sum_{j=1}^{p} \Big(A_i b_j\Big)^2, \tag{23}$$

where c_{ij} denotes the (i,j)-th entry of C, A_i is the i-th row of A, and b_j is the j-th column of B. By Cauchy-Schwarz Inequality Bityutskov (2001), we obtain that

$$\begin{aligned} ||C||_F^2 &\le \sum_{i=1}^{m}\sum_{j=1}^{p} ||A_i||^2||b_j||^2 \\ &= \Big(\sum_{i=1}^{m}||A_i||^2\Big)\Big(\sum_{j=1}^{p}||b_j||^2\Big) \\ &= ||A||_F^2||B||_F^2 \end{aligned} \tag{24}$$

Taking the square root of each side completes the proof. □

Theorem 2 (Restatement)**.** *Let $D^1 \cdots D^N \in \mathbb{R}^{d\times d}$ be a Deep Linear Projector. Then, the following inequality holds:*

$$||D^1 \cdots D^N||_* \le \Big(\frac{1}{N}\cdot\sum_{n=1}^{N}||D^n||_F\Big)^N, \tag{25}$$

where $||\cdot||_*$ is the nuclear norm and $||\cdot||_F$ is the Frobenius norm.

Proof. Let $D = D^1 \cdots D^N$ be a deep linear projector and abbreviate $\hat{D} := D^2 \cdots D^N$. By Hölder's inequality for Schatten norms, the nuclear norm of D is given by

$$\|D\|_* = \|D^1\hat{D}\|_* \le \|D^1\|_F\|\hat{D}\|_F \tag{26}$$

By Lemma 1, we can derive the following:

$$||D^1||_F||\hat{D}||_F \le ||D^1||_F \cdots ||D^N||_F \tag{27}$$

Using the AM-GM Inequality, we can derive the following inequality:

$$\begin{aligned} ||D^1||_F \cdots ||D^N||_F &\le \Big(\frac{1}{N}\cdot(||D^1||_F + \cdots + ||D^N||_F)\Big)^N \\ &= \Big(\frac{1}{N}\cdot\sum_{n=1}^{N}||D^n||_F\Big)^N \end{aligned} \tag{28}$$

Combining Eq. (26) - (28) yields the following inequality:

$$||D||_* \le \Big(\frac{1}{N}\cdot\sum_{n=1}^{N}||D^n||_F\Big)^N \tag{29}$$

□

C Training and Implementation Details

Common Settings. All experiments use AdamW with a warm-up ratio of 0.03 followed by learning-rate decay, bfloat16 training, and DeepSpeed ZeRO Stage 2. Unless otherwise specified, runs use 4×H100 (80GB) GPUs, a global batch size of 32, and a maximum sequence length of 2048.

Stage 1. We use $N = 2$, $d_{\text{hidden}} = 256$, and $\alpha = 0.05$. Full-dimensional DLPs are initialized as identity mappings and trained for one epoch with a peak learning rate of 5×10^{-4} on 100K samples drawn from SlimPajama-6B.

Stage 2. After rank selection, the corresponding boundary factors are reduced by SVD while the remaining Stage 1 factors are retained. Stage 2 therefore starts directly from the reduced Stage 1 factors rather than reinitializing the DLRPs. Recovery fine-tuning uses 20K SlimPajama-6B samples for one epoch with a learning rate of 5×10^{-4}.

C.1 RoPE Compatibility

For RoPE-based models, Stage 1 is performed in the original d_{head}-dimensional space with RoPE unchanged. After dimensionality reduction, RoPE is applied in the reduced query/key dimension r^l_{qk}, and scaled dot-product attention uses the factor $1/\sqrt{r^l_{\text{qk}}}$. This keeps positional encoding consistent with the dimensionality used by the reduced query and key projections.

C.2 Variable-Rank Triton Attention Kernel

Layer-Wise Asymmetric Dimensions. DLRP allows the reduced query–key and value–output dimensions to differ across layers. For layer l, we denote these dimensions by r^l_{qk} and r^l_{vo}, respectively. The pair $[r^l_{\text{qk}}, r^l_{\text{vo}}]$ is stored for each layer in the model configuration. The dimensions vary across layers, while all attention heads within the same layer share the same pair. Thus, DLRP uses layer-wise rather than head-wise variable dimensions.

Conceptually, the reduced projections can be written as

$$Q' = QP_Q, \qquad K' = KP_K, \qquad V' = VP_V, \tag{30}$$

followed by

$$O = \text{Attn}(Q', K', V')P_O, \tag{31}$$

where

$$Q', K' \in \mathbb{R}^{r^l_{\text{qk}}}, \qquad V', \text{Attn}(Q', K', V') \in \mathbb{R}^{r^l_{\text{vo}}}. \tag{32}$$

After recovery, these projections are folded into the original attention weights, so no additional projection modules remain in the inference path.

The resulting attention operation is

$$\text{Attn}(Q', K', V') = \text{Softmax}\left(\frac{Q'K'^{\top}}{\sqrt{r^l_{\text{qk}}}}\right)V'. \tag{33}$$

Thus, the score reduction and scaling are determined by r^l_{qk}, whereas the value and attention-output dimension is r^l_{vo}.

Layer-Wise Rank Selection. For each projection, we select the minimum dimension satisfying the singular-value energy criterion in Eq. (9). The dimensions used by attention are then

$$r^l_{\text{qk}} = \max(r^l_{\text{query}}, r^l_{\text{key}}), \qquad r^l_{\text{vo}} = \max(r^l_{\text{value}}, r^l_{\text{output}}). \tag{34}$$

The implementation does not require the selected dimensions to be powers of two or multiples of eight, allowing arbitrary integer dimensions across layers.

Asymmetric Triton Kernel. Standard FlashAttention-2 assumes matching query, key, and value head dimensions. Since DLRP can produce $r^l_{\text{qk}} \neq r^l_{\text{vo}}$, we implement a forward asymmetric attention kernel in Triton. Its inputs and output have shapes

$$\begin{aligned} Q &\in \mathbb{R}^{B\times H\times M\times D_K}, \\ K &\in \mathbb{R}^{B\times H_{\text{KV}}\times N\times D_K}, \\ V &\in \mathbb{R}^{B\times H_{\text{KV}}\times N\times D_V}, \\ O &\in \mathbb{R}^{B\times H\times M\times D_V}, \end{aligned} \tag{35}$$

where

$$D_K = r^l_{\text{qk}}, \qquad D_V = r^l_{\text{vo}}. \tag{36}$$

For each query tile, the kernel loads a $\mathrm{BLOCK}_M \times \mathrm{BLOCK}_{DK}$ query block and a $\mathrm{BLOCK}_N \times \mathrm{BLOCK}_{DK}$ key block to compute the attention scores using the scaling factor $1/\sqrt{D_K}$. It applies online softmax without materializing the full attention matrix. A $\mathrm{BLOCK}_N \times \mathrm{BLOCK}_{DV}$ value block is then multiplied by the normalized attention probabilities and accumulated into a $\mathrm{BLOCK}_M \times \mathrm{BLOCK}_{DV}$ output buffer. Consequently, the score-reduction dimension D_K and the value/output dimension D_V are handled independently.

The internal block dimensions are selected as

$$\mathrm{BLOCK}_{DK} = \mathrm{next_power_of_2}(D_K), \qquad \mathrm{BLOCK}_{DV} = \mathrm{next_power_of_2}(D_V), \tag{37}$$

with masking applied outside the actual dimensions. Hence, D_K and D_V themselves need not be powers of two or multiples of eight. We use $\mathrm{BLOCK}_N = 64$ and query tiles of up to 128 tokens.

The kernel also supports grouped-query attention (GQA). Query head h is mapped to its corresponding KV head as

$$h_{\mathrm{KV}} = \left\lfloor \frac{h H_{\mathrm{KV}}}{H} \right\rfloor. \tag{38}$$

For causal generation with $M \neq N$, the absolute position of query index i is computed as

$$N - M + i, \tag{39}$$

which is used to construct the causal mask.

Execution Modes. We support two execution paths when $r^l_{\mathrm{qk}} \neq r^l_{\mathrm{vo}}$. In the *pad* mode, Q, K, and V are zero-padded to $\max(r^l_{\mathrm{qk}}, r^l_{\mathrm{vo}})$ and evaluated using the standard FlashAttention-2 backend. This path serves as a correctness reference and fallback. In the *triton* mode, the asymmetric kernel operates directly on the actual D_K and D_V dimensions without zero-padding. When $r^l_{\mathrm{qk}} = r^l_{\mathrm{vo}}$, the existing attention backend is used directly.

RULER Evaluation. For RULER, DLRP checkpoints are loaded with the custom Qwen3 implementation and the Triton execution mode is enabled for layers with $r^l_{\mathrm{qk}} \neq r^l_{\mathrm{vo}}$. We evaluate context lengths of 4K, 8K, 16K, and 32K with batch size 1 and bfloat16 precision. The base attention backend is FlashAttention-2, and generation uses greedy decoding with no few-shot examples and a maximum generation length of 128 tokens.

The current Triton implementation provides a forward kernel for variable-rank inference. Training and recovery fine-tuning do not rely on a corresponding custom backward kernel.

D Additional Main Results

D.1 Complete Short-Context Results

We provide the complete task-wise short-context results across model families and compression ratios. The Qwen3 results are shown in Table 8, and the Llama results are shown in Table 9.

Table 8: Complete zero-shot short-context results for Qwen3-4B-Base and Qwen3-8B-Base under different KV-cache compression ratios (CR).

Model	CR	PIQA	ARC-C	ARC-E	HSwag	Wino	TQA	Avg.
Qwen3-4B	–	77.80	47.60	80.10	61.30	72.40	44.68	63.98
	23.2%	77.81	47.71	80.03	61.28	72.43	44.62	63.98
	27.3%	77.53	47.24	79.86	61.02	72.13	41.84	63.27
	30.3%	77.20	46.90	79.60	60.60	71.90	40.72	62.82
	32.9%	76.71	45.98	78.91	59.84	71.18	38.72	61.89
Qwen3-8B	–	80.40	53.70	84.00	64.80	75.80	39.46	66.36
	26.3%	80.61	54.02	84.16	64.96	76.01	40.86	66.77
	30.1%	80.28	53.62	83.94	64.73	75.69	39.54	66.30
	32.4%	80.00	53.20	83.70	64.40	75.40	39.84	66.09
	35.6%	79.41	52.31	82.92	63.72	74.61	37.81	65.13

Table 9: Complete zero-shot short-context results for Llama-3.2-3B and Llama-3.1-8B under different KV-cache compression ratios (CR).

Model	CR	PIQA	ARC-C	ARC-E	HSwag	Wino	TQA	Avg.
Llama-3.2-3B	–	76.61	42.83	74.49	55.83	69.06	51.47	61.72
	23.2%	76.60	43.00	74.80	55.60	69.00	51.35	61.73
	27.3%	76.10	42.70	74.40	54.80	68.85	51.00	61.31
	30.3%	75.85	42.30	74.20	54.30	68.65	50.50	60.97
	32.9%	75.00	41.00	73.30	52.50	67.60	49.20	59.77
Llama-3.1-8B	–	79.05	52.13	82.24	60.71	74.59	44.16	65.48
	26.3%	78.92	52.03	82.14	60.64	74.34	44.02	65.35
	30.1%	78.68	51.93	81.92	60.31	74.10	43.78	65.12
	32.4%	78.53	51.82	81.60	60.12	73.95	43.48	64.92
	35.6%	78.19	50.92	80.02	58.14	71.88	41.02	63.36

D.2 Complete LongBench Results

Tables 10 and 11 report category-wise LongBench results. We report Single-Document QA, Multi-Document QA, Summarization (Summ), Few-shot Learning (FSL), Synthetic tasks (Synth), and their average.

Table 10: Complete LongBench results for Qwen3-4B-Base and Qwen3-8B-Base.

Model	CR	Single	Multi	Summ	FSL	Synth	Avg.
Qwen3-4B	–	40.59	38.42	27.41	69.53	43.50	43.89
	23.2%	40.58	38.32	27.40	69.49	43.43	43.84
	27.3%	40.14	37.87	26.95	68.81	42.99	43.35
	30.3%	39.84	37.61	26.80	68.35	42.78	43.07
	32.9%	38.69	36.41	25.82	66.36	41.80	41.81
Qwen3-8B	–	36.53	42.75	27.48	69.78	48.33	44.97
	26.3%	36.45	42.62	27.40	69.65	48.25	44.87
	30.1%	36.19	42.25	27.20	69.24	48.00	44.58
	32.4%	35.98	41.95	27.02	68.88	47.78	44.32
	35.6%	35.08	40.69	26.28	67.44	46.97	43.29

Table 11: Complete LongBench results for Llama-3.2-3B and Llama-3.1-8B.

Model	CR	Single	Multi	Summ	FSL	Synth	Avg.
Llama-3.2-3B	–	12.04	2.64	21.06	63.65	1.67	20.21
	23.2%	12.05	2.64	20.76	63.53	1.67	20.13
	27.3%	11.97	2.62	20.93	63.23	1.66	20.08
	30.3%	11.90	2.61	20.81	62.88	1.65	19.97
	32.9%	11.66	2.55	20.40	61.64	1.61	19.57
Llama-3.1-8B	–	13.19	3.31	16.09	68.88	2.97	20.89
	26.3%	13.17	3.30	16.05	68.73	2.97	20.84
	30.1%	13.11	3.28	15.97	68.48	2.95	20.76
	32.4%	13.07	3.27	15.91	68.27	2.95	20.69
	35.6%	12.72	3.17	15.44	66.60	2.87	20.16

D.3 Complete Token-Axis Compatibility Results

We provide detailed results for combining DLRP with token-axis compression. For each token-axis method, we evaluate 60% and 80% token compression and combine it with three DLRP compression levels. This evaluates whether the two compression axes can be applied jointly without introducing a large additional degradation.

Table 12: Complete LongBench compatibility results for token-axis compression and DLRP on Qwen3-4B-Base.

Method	Setting	Single	Multi	Summ	FSL	Synth	Avg.
Backbone	–	40.59	38.42	27.41	69.53	43.50	43.89
SnapKV	60%	39.78	37.12	26.84	67.55	42.13	42.68
SnapKV + DLRP	60% + 23.2%	39.70	37.04	26.80	67.43	42.04	42.60
SnapKV + DLRP	60% + 27.3%	39.50	36.83	26.66	67.08	41.82	42.38
SnapKV + DLRP	60% + 30.3%	39.28	36.59	26.50	66.71	41.56	42.13
SnapKV	80%	38.42	35.90	25.81	64.88	40.72	41.15
SnapKV + DLRP	80% + 23.2%	38.35	35.82	25.77	64.77	40.63	41.07
SnapKV + DLRP	80% + 27.3%	38.13	35.58	25.61	64.42	40.38	40.82
SnapKV + DLRP	80% + 30.3%	37.90	35.34	25.45	64.02	40.11	40.56
StreamingLLM	60%	31.84	29.67	22.10	54.90	31.44	33.99
StreamingLLM + DLRP	60% + 23.2%	31.77	29.61	22.06	54.80	31.37	33.92
StreamingLLM + DLRP	60% + 27.3%	31.60	29.44	21.94	54.49	31.17	33.73
StreamingLLM + DLRP	60% + 30.3%	31.40	29.25	21.81	54.12	30.94	33.50
StreamingLLM	80%	29.24	27.10	20.41	50.30	28.92	31.19
StreamingLLM + DLRP	80% + 23.2%	29.18	27.05	20.37	50.20	28.86	31.13
StreamingLLM + DLRP	80% + 27.3%	29.02	26.89	20.26	49.91	28.66	30.95
StreamingLLM + DLRP	80% + 30.3%	28.84	26.71	20.14	49.58	28.45	30.74
KeyDiff	60%	33.88	32.74	23.16	58.61	34.44	36.57
KeyDiff + DLRP	60% + 23.2%	33.82	32.67	23.12	58.49	34.36	36.49
KeyDiff + DLRP	60% + 27.3%	33.64	32.49	22.99	58.16	34.15	36.29
KeyDiff + DLRP	60% + 30.3%	33.45	32.28	22.86	57.80	33.91	36.06
KeyDiff	80%	30.74	29.38	21.17	52.48	30.69	32.89
KeyDiff + DLRP	80% + 23.2%	30.68	29.32	21.13	52.38	30.62	32.83
KeyDiff + DLRP	80% + 27.3%	30.52	29.16	21.02	52.10	30.43	32.65
KeyDiff + DLRP	80% + 30.3%	30.35	28.98	20.90	51.78	30.22	32.45
PyramidKV	60%	39.55	36.91	26.63	66.88	41.92	42.38
PyramidKV + DLRP	60% + 23.2%	39.49	36.84	26.59	66.77	41.85	42.31
PyramidKV + DLRP	60% + 27.3%	39.32	36.66	26.46	66.48	41.65	42.11
PyramidKV + DLRP	60% + 30.3%	39.14	36.47	26.34	66.13	41.43	41.90
PyramidKV	80%	37.92	35.49	25.40	63.84	39.98	40.53
PyramidKV + DLRP	80% + 23.2%	37.87	35.43	25.36	63.75	39.91	40.46
PyramidKV + DLRP	80% + 27.3%	37.71	35.27	25.25	63.48	39.74	40.29
PyramidKV + DLRP	80% + 30.3%	37.54	35.09	25.13	63.18	39.53	40.09

Table 13: Complete LongBench compatibility results for token-axis compression and DLRP on Qwen3-8B-Base.

Method	Setting	Single	Multi	Summ	FSL	Synth	Avg.
Backbone	–	36.53	42.75	27.48	69.78	48.33	44.97
SnapKV	60%	35.66	41.34	26.82	67.95	47.93	43.94
SnapKV + DLRP	60% + 26.3%	35.54	41.18	26.74	67.72	47.75	43.79
SnapKV + DLRP	60% + 30.1%	35.30	40.93	26.50	67.20	47.38	43.46
SnapKV + DLRP	60% + 32.4%	35.18	40.71	26.32	66.94	47.45	43.32
SnapKV	80%	34.76	39.92	25.74	64.80	47.83	42.61
SnapKV + DLRP	80% + 26.3%	34.62	39.75	25.63	64.54	47.66	42.44
SnapKV + DLRP	80% + 30.1%	34.42	39.49	25.42	64.10	47.35	42.16
SnapKV + DLRP	80% + 32.4%	34.30	39.31	25.30	63.88	47.61	42.08
StreamingLLM	60%	29.93	32.14	22.41	50.35	45.12	35.99
StreamingLLM + DLRP	60% + 26.3%	29.85	32.05	22.34	50.19	45.02	35.89
StreamingLLM + DLRP	60% + 30.1%	29.69	31.87	22.21	49.88	44.80	35.69
StreamingLLM + DLRP	60% + 32.4%	29.48	31.68	22.08	49.54	44.37	35.43
StreamingLLM	80%	28.14	30.02	20.88	47.05	43.11	33.84
StreamingLLM + DLRP	80% + 26.3%	28.06	29.93	20.81	46.89	43.01	33.74
StreamingLLM + DLRP	80% + 30.1%	27.91	29.76	20.69	46.61	42.82	33.56
StreamingLLM + DLRP	80% + 32.4%	27.75	29.58	20.57	46.32	42.48	33.34
KeyDiff	60%	31.42	33.58	23.07	55.31	30.52	34.78
KeyDiff + DLRP	60% + 26.3%	31.34	33.50	23.00	55.17	30.43	34.69
KeyDiff + DLRP	60% + 30.1%	31.18	33.31	22.87	54.84	30.23	34.49
KeyDiff + DLRP	60% + 32.4%	31.02	33.14	22.75	54.56	30.28	34.35
KeyDiff	80%	29.24	31.10	21.22	51.04	30.00	32.52
KeyDiff + DLRP	80% + 26.3%	29.16	31.02	21.16	50.90	29.91	32.43
KeyDiff + DLRP	80% + 30.1%	29.02	30.85	21.04	50.60	29.72	32.25
KeyDiff + DLRP	80% + 32.4%	28.88	30.69	20.93	50.34	29.66	32.10
PyramidKV	60%	35.82	40.98	26.51	67.06	46.53	43.38
PyramidKV + DLRP	60% + 26.3%	35.74	40.88	26.45	66.92	46.45	43.29
PyramidKV + DLRP	60% + 30.1%	35.59	40.72	26.33	66.62	46.26	43.10
PyramidKV + DLRP	60% + 32.4%	35.48	40.59	26.24	66.39	46.90	43.12
PyramidKV	80%	34.71	39.41	25.61	64.40	45.47	41.92
PyramidKV + DLRP	80% + 26.3%	34.64	39.32	25.55	64.27	45.38	41.83
PyramidKV + DLRP	80% + 30.1%	34.50	39.17	25.44	63.99	45.19	41.66
PyramidKV + DLRP	80% + 32.4%	34.39	39.04	25.35	63.77	45.60	41.63

Table 14: Complete LongBench compatibility results for token-axis compression and DLRP on Llama-3.2-3B and Llama-3.1-8B. For DLRP compression settings, values before and after the slash correspond to Llama-3.2-3B and Llama-3.1-8B, respectively.

		Llama-3.2-3B						Llama-3.1-8B					
Method	**Setting**	**Single**	**Multi**	**Summ**	**FSL**	**Synth**	**Avg.**	**Single**	**Multi**	**Summ**	**FSL**	**Synth**	**Avg.**
Backbone	–	12.04	2.64	21.06	63.65	1.67	20.21	13.19	3.31	16.09	68.88	2.97	20.89
SnapKV	60%	10.67	2.61	17.19	59.16	2.09	18.34	11.42	3.40	13.24	65.05	2.57	19.14
SnapKV + DLRP	60% + 23.2% / 26.3%	10.67	2.61	17.18	59.14	2.09	18.34	11.34	3.38	13.15	64.63	2.55	19.01
SnapKV + DLRP	60% + 27.3% / 30.1%	10.60	2.59	17.07	58.77	2.07	18.22	11.25	3.35	13.04	64.10	2.53	18.85
SnapKV + DLRP	60% + 30.3% / 32.4%	10.54	2.57	16.98	58.45	2.06	18.12	11.16	3.32	12.94	63.58	2.51	18.70
SnapKV	80%	10.62	2.61	15.77	53.81	2.09	16.98	12.66	3.11	14.73	63.66	2.58	19.35
SnapKV + DLRP	80% + 23.2% / 26.3%	10.62	2.61	15.77	53.79	2.09	16.98	12.53	3.08	14.57	63.02	2.55	19.15
SnapKV + DLRP	80% + 27.3% / 30.1%	10.55	2.59	15.67	53.45	2.07	16.87	12.43	3.06	14.45	62.51	2.53	19.00
SnapKV + DLRP	80% + 30.3% / 32.4%	10.49	2.58	15.58	53.16	2.06	16.77	12.32	3.03	14.33	62.00	2.51	18.84
StreamingLLM	60%	6.02	2.04	9.87	39.88	2.79	12.12	6.92	1.96	9.85	41.51	4.17	12.88
StreamingLLM + DLRP	60% + 23.2% / 26.3%	6.01	2.04	9.86	39.83	2.78	12.10	6.87	1.94	9.78	41.18	4.13	12.78
StreamingLLM + DLRP	60% + 27.3% / 30.1%	5.98	2.03	9.81	39.62	2.77	12.04	6.82	1.93	9.70	40.83	4.10	12.68
StreamingLLM + DLRP	60% + 30.3% / 32.4%	5.96	2.02	9.77	39.49	2.76	12.00	6.76	1.91	9.62	40.50	4.07	12.57
StreamingLLM	80%	5.20	1.60	9.41	35.97	1.85	10.81	7.17	2.54	11.63	40.21	3.40	12.99
StreamingLLM + DLRP	80% + 23.2% / 26.3%	5.19	1.60	9.40	35.92	1.85	10.79	7.09	2.51	11.50	39.73	3.36	12.84
StreamingLLM + DLRP	80% + 27.3% / 30.1%	5.16	1.59	9.35	35.73	1.84	10.73	7.03	2.49	11.41	39.40	3.33	12.73
StreamingLLM + DLRP	80% + 30.3% / 32.4%	5.14	1.59	9.32	35.62	1.84	10.70	6.98	2.47	11.31	39.08	3.31	12.63
KeyDiff	60%	10.27	2.68	18.27	53.36	2.08	17.33	10.71	3.27	13.50	61.59	2.56	18.33
KeyDiff + DLRP	60% + 23.2% / 26.3%	10.25	2.68	18.24	53.28	2.08	17.31	10.64	3.24	13.39	61.13	2.54	18.19
KeyDiff + DLRP	60% + 27.3% / 30.1%	10.20	2.66	18.15	53.01	2.07	17.22	10.55	3.21	13.28	60.62	2.52	18.04
KeyDiff + DLRP	60% + 30.3% / 32.4%	10.16	2.65	18.09	52.84	2.06	17.16	10.46	3.19	13.17	60.13	2.50	17.89
KeyDiff	80%	8.21	2.72	16.73	41.71	1.67	14.21	10.08	2.89	13.17	59.00	3.02	17.63
KeyDiff + DLRP	80% + 23.2% / 26.3%	8.20	2.72	16.71	41.65	1.67	14.19	9.97	2.85	13.01	58.32	2.99	17.43
KeyDiff + DLRP	80% + 27.3% / 30.1%	8.16	2.70	16.62	41.43	1.66	14.11	9.89	2.83	12.91	57.83	2.96	17.28
KeyDiff + DLRP	80% + 30.3% / 32.4%	8.13	2.69	16.56	41.30	1.65	14.07	9.80	2.81	12.80	57.36	2.94	17.14
PyramidKV	60%	10.15	2.37	16.82	57.40	2.08	17.76	11.71	3.14	15.64	66.43	2.58	19.90
PyramidKV + DLRP	60% + 23.2% / 26.3%	10.13	2.37	16.44	57.30	2.08	17.66	11.65	3.13	15.57	66.09	2.57	19.80
PyramidKV + DLRP	60% + 27.3% / 30.1%	10.09	2.35	16.37	57.06	2.07	17.59	11.58	3.11	15.47	65.69	2.55	19.68
PyramidKV + DLRP	60% + 30.3% / 32.4%	10.05	2.35	16.31	56.86	2.06	17.53	11.51	3.09	15.38	65.30	2.54	19.56
PyramidKV	80%	9.77	2.61	14.86	54.48	2.08	16.76	11.74	3.04	14.24	65.13	2.60	19.35
PyramidKV + DLRP	80% + 23.2% / 26.3%	9.75	2.61	14.84	54.39	2.08	16.73	11.67	3.03	14.15	64.72	2.59	19.23
PyramidKV + DLRP	80% + 27.3% / 30.1%	9.71	2.60	14.78	54.16	2.07	16.66	11.60	3.01	14.07	64.33	2.57	19.12
PyramidKV + DLRP	80% + 30.3% / 32.4%	9.68	2.59	14.73	53.97	2.06	16.61	11.53	2.99	13.98	63.94	2.56	19.00

E Additional Ablation Studies

E.1 Effect of Regularization Strength

We vary α while fixing the energy threshold at $e = 0.999$. Moderate regularization provides the best overall trade-off, while stronger regularization produces smaller selected dimensions with a larger performance drop.

Table 15: Sensitivity to the regularization strength α on Qwen3-4B-Base.

α	0.001	0.01	0.05	0.1	0.5
Short Avg.	63.70	63.90	**63.98**	63.40	63.10
LongBench	43.50	43.80	**43.84**	43.30	42.90

E.2 Additional Two-Stage Design Results

To separate the effect of Stage 1 from the final reduced architecture, the direct baseline uses exactly the same layer-wise dimensions as the full DLRP but trains the reduced projectors from scratch. Table 16 extends this comparison across multiple compression ratios.

Table 16: Two-stage ablation across compression ratios.

Model	**CR**	**Direct reduced**	**After reduction**	**Full DLRP**
Qwen3-4B	23.2%	62.91	63.12	63.98
	27.3%	62.35	62.51	63.27
	30.3%	61.68	61.94	62.82
Qwen3-8B	26.3%	65.55	65.91	66.77
	30.1%	65.11	65.62	66.30
	32.4%	64.95	65.42	66.09

E.3 Additional Layer-Wise Rank Allocation Results

We compare uniform, random, and spectrum-based allocation under the same total reduced-dimension budget. Random allocation is averaged over five independently sampled assignments.

Table 17: Rank-allocation ablation across compression ratios.

Model	CR	Uniform	Random	Spectrum
Qwen3-4B	23.2%	63.12	62.86±0.20	63.98
	27.3%	62.57	62.11±0.24	63.27
	30.3%	61.96	61.43±0.27	62.82
Qwen3-8B	26.3%	66.12	65.90±0.18	66.77
	30.1%	65.84	65.55±0.22	66.30
	32.4%	65.61	65.18±0.24	66.09

E.4 Boundary Reduction Strategy

We compare alternative ways of constructing the reduced DLRP after rank selection. Reinitialization discards the trained Stage 1 factors, direct slicing keeps the corresponding leading coordinates, and the proposed strategy reduces the trained boundary factor in the SVD domain while retaining the remaining Stage 1 factors.

Table 18: Boundary-reduction strategies at approximately 30% compression.

Strategy	Qwen3-4B	Qwen3-8B
Reinitialize reduced DLRP	61.68	64.95
Direct slicing	62.14	65.47
Boundary-factor SVD	**62.82**	**66.30**

E.5 DLP Architecture

Although the final composed DLP is linear, its factorized parameterization changes the optimization trajectory. Table 19 compares the final Stage 1 training loss under different DLP depths.

Table 19: Final Stage 1 training loss under different DLP depths on Qwen3-4B-Base.

Depth	$N = 1$	$N = 2$	$N = 3$
Final training loss	0.7011	**0.4917**	0.5142

E.6 Dataset Size

We vary the recovery-set size from 5K to 100K samples. Performance improves up to approximately 20K samples and then saturates, indicating that Stage 2 does not require large-scale retraining.

Table 20: Short-context performance on Qwen3-4B-Base under different recovery-set sizes.

Samples	5K	10K	20K	50K	100K	Backbone
Short Avg.	60.50	62.32	**63.98**	63.97	64.02	63.98

F Additional Analysis

F.1 Regularization and Spectral Structure

The theorem in Section 4.1 provides an upper-bound relation rather than a guarantee that the nuclear norm decreases monotonically during optimization. We therefore examine the empirical rank structure produced by the regularizer. Figure 4 shows the average selected dimensions under a fixed energy threshold. Stronger regularization leads to lower selected dimensions for both query–key and value–output paths. This trend is consistent with the intended role of the upper-bound surrogate: the learned spectra can be represented with fewer singular components under the same energy criterion.

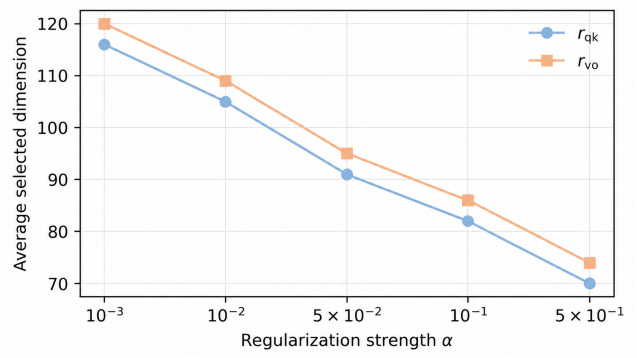

Figure 4: Average selected dimensions under different regularization strengths at a fixed energy threshold.

F.2 Layer-Wise Rank Distribution

The selected dimensions are not uniform across layers, and the query–key and value–output paths also exhibit different layer-wise profiles. This provides additional motivation for spectrum-based allocation instead of assigning a single reduced dimension to every layer. The non-uniformity becomes more pronounced as the energy threshold is lowered, reflecting different spectral concentration across layers.

F.3 Stage-Wise Recovery

Figure 5 separates the effect of dimensionality reduction from Stage 2 recovery. Reducing the trained Stage 1 factors causes an immediate performance drop, while Stage 2 recovers a substantial part of this loss. This supports the use of Stage 1 parameters as the initialization of the reduced DLRP rather than treating Stage 1 only as a rank-selection mechanism.

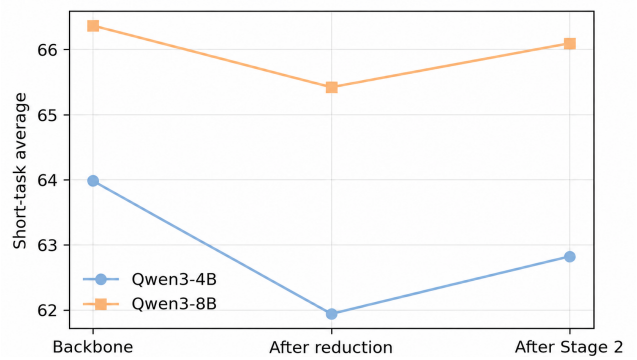

Figure 5: Short-context performance before and after dimensionality reduction and Stage 2 recovery at the representative compression setting.

F.4 Compression–Performance Trend

Across the four model families, performance decreases gradually as the selected dimensions become smaller. Figure 6 summarizes this trade-off across model scales.

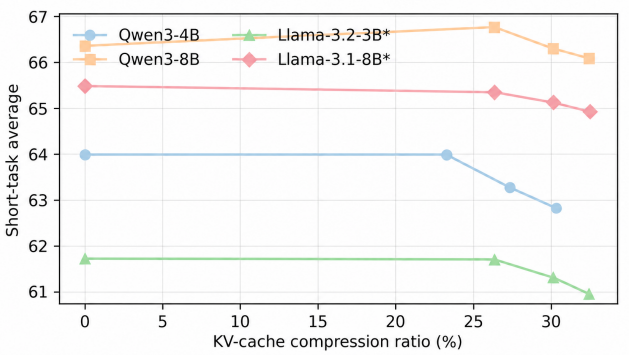

Figure 6: Short-context performance across KV-cache compression ratios.

F.5 Serving Efficiency Across Context Lengths

The main paper reports serving measurements at 16K and 64K. DLRP reduces peak memory and increases the maximum feasible batch size at both lengths, while the current Triton implementation has lower decoding throughput at 64K. This suggests that the memory benefit of reduced KV representations is already realized, whereas variable-rank kernel efficiency remains an implementation-level optimization opportunity.

Table 21: Serving efficiency on Qwen3-8B-Base with the variable-rank Triton kernel.

Model	Context	Max BS	VRAM	Tok./s
Backbone	16K	14	69.65 GB	83.8
DLRP	16K	23	65.11 GB	87.2
Backbone	64K	3	61.89 GB	14.4
DLRP	64K	5	58.64 GB	11.0