Low-Rank Key Value Attention

Shared projections, specialized heads, and the cost of a smaller cache

Paper authors: James O'Neill, Robert Clancy,
Mariia Matskevichus, Fergal Reid

Independent paper discussion

arXiv:2601.11471v3 (7 April 2026)
Reading date: 15 October 2026

A shared K/V base plus per-head low-rank corrections can reduce cache storage.
The deployment claims require closer scrutiny than the linear algebra.

Can heads share storage while keeping distinct K/V features?

The design question

How much of each head's key/value representation must be stored independently?

Mechanism	K/V structure	Cache elements / token / layer
MHA	Independent per head	$2Hd_h$
MQA	One shared pair	$2d_h$
GQA	n_{kv} shared groups	$2n_{kv}d_h$
LRKV	Shared base + residuals	$2(d_h + Hr)$

- H: query heads; d_h: head width; r: residual rank.
- Queries remain independent in all four designs.

Paper, Secs. 1–3, PDF pp. 1–4. Cache excludes model parameters.

LRKV constrains differences between heads

$$W_h^{K,V} = W_{\text{shared}}^{K,V} + U_h^{K,V}(B_h^{K,V})^\top$$

Shared component	Head-specific correction
$W_{\text{shared}} \in \mathbb{R}^{d \times d_h}$	$U_h \in \mathbb{R}^{d \times r}$, $B_h \in \mathbb{R}^{d_h \times r}$
Dense, with no rank constraint.	$\text{rank}(U_h B_h^\top) \leq r$.
One common projection across heads.	Separate factors for keys and values.

The effective W_h may be full rank, while the family of heads is constrained.

Paper, Sec. 3, Eqs. (2)–(3), PDF p. 3. Here $d = Hd_h$.

Training learns the base and residuals together

- Jointly train W_{shared}, U_h, and B_h from scratch.
- Initialize residuals near the shared baseline: about $0.1\|W_{\text{shared}}\|_F$.
- Keep full, independent query projections W_h^Q.
- Materialize full per-head K/V during training.

Two different notions of equivalence

Factored and expanded evaluation of the *same LRKV model* are equivalent in the linear setting. A low-rank LRKV model does not represent every possible MHA parameterization.

Paper, Sec. 3, training and initialization, PDF p. 3. Cache savings target decoding.

Cache shared features and compact per-head latents

Once per layer

$$K_s = XW^K_{\text{shared}}, \qquad V_s = XW^V_{\text{shared}}$$

Each tensor: $L \times d_h$.

For each head

$$R^K_h = XU^K_h, \quad R^V_h = XU^V_h$$

Each tensor: $L \times r$.

The implied full features are

$$K_h = K_s + R^K_h (B^K_h)^\top, \qquad V_h = V_s + R^V_h (B^V_h)^\top.$$

- L: cached sequence length; $X \in \mathbb{R}^{L \times d}$.
- $B^{K,V}_h$ are weights, not per-token cache entries.

Paper, Sec. 3, Eqs. (4)–(5), PDF p. 3. K_s, V_s abbreviate the shared features.

Compute logits without expanding historical keys

For the current query $q_h \in \mathbb{R}^{1\times d_h}$:

$$\underbrace{q_h K_h^\top}_{1\times L} = \underbrace{q_h K_s^\top}_{\text{shared match}} + \underbrace{(q_h B_h^K)(R_h^K)^\top}_{\text{head-specific correction}} .$$

1. Project the *current query* into r dimensions.
2. Compare it with the cached residual latents.
3. Add both score paths, then apply scaling and softmax.

$$a_h = \text{softmax}\left(\frac{q_h K_h^\top}{\sqrt{d_h}} + \text{causal mask}\right).$$

Paper, Eq. (6), PDF p. 4. Exact for the stated linear projection model.

Aggregate values before the residual up-projection

For attention weights $a_h \in \mathbb{R}^{1\times L}$:

$$\boxed{o_h = a_h V_s + (a_h R_h^V)(B_h^V)^\top}$$

Shared path

$a_h V_s$ produces one d_h-dimensional output.

Residual path

First reduce L latents to one r-vector; then map that vector to d_h.

- Historical per-head K/V need not be materialized.
- Dense softmax and the scan over past tokens remain.

Paper, Eq. (7), PDF p. 4. Position encoding and normalization require additional treatment.

The rank directly sets the theoretical cache budget

$$\rho = \frac{2L(d_h + Hr)}{2LHd_h} = \boxed{\frac{1}{H} + \frac{r}{d_h}}.$$

Scale label	H	r (Table 7)	Recomputed cache / MHA
128M	6	46	52.60%
1.2B	12	51	48.18%
2.5B	18	55	48.52%
6.3B	32	54	45.31%

All rows use $d_h = 128$. At 2.5B: 4,608 $\rightarrow$ 2,236 elements per token per layer.

- $r = 0$: MQA-style sharing.
- $r = d_h$: expressive enough for MHA, but cache ratio is $1 + 1/H$.

Paper, Eq. (9), p. 4; Table 7, p. 24. Percentages recomputed from those configurations.

Smaller storage trades bandwidth for extra arithmetic

Per head, per decoding step

MHA's main scan: $O(Ld_h)$ LRKV's additional work: $O(Lr + rd_h)$.

- The residual scan costs roughly r/d_h of the main attention scan at long context.
- Shared cache reuse may reduce memory traffic; actual latency depends on the kernel.
- Cache savings and K/V parameter savings have different formulas:

$$P_{\mathrm{LRKV,KV}} = 2dd_h + 2Hr(d + d_h), \quad P_{\mathrm{MHA,KV}} = 2Hdd_h.$$

Paper, Sec. 3 and Eq. (10), PDF p. 4. No end-to-end decoding latency table is provided.

The experiments change attention within a shared training recipe

Scale label	Context	Tokens	Role
128M	2,048	100B	Main comparison + rank sweep
1.2B / 2.5B / 6.3B	2,048	50B each	Scaling comparison
512M	8,192	50B	Long-context training

- FineWeb-Edu; 8 H200 GPUs; bfloat16; Muon + AdamW.
- Baselines: MHA, GQA, MQA, and MLA, with different cache budgets.
- Scale labels refer to the MHA base, not equal total parameters.
- Downstream: one epoch of supervised midtraining on 568K examples.

Paper, Sec. 4; Appendix D, Tables 5–7, PDF pp. 23–24. Architecture inconsistencies discussed later.

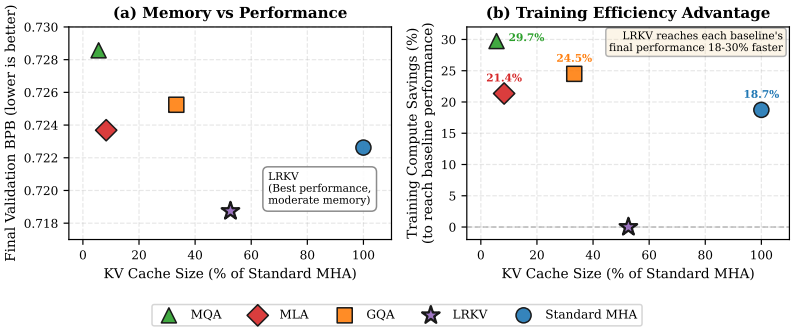
Table 1 reports the lowest final CE at every scale

Method	128M	1.2B	2.5B	6.3B
MHA	2.903	2.530	2.389	2.319
GQA	2.918	2.536	2.397	2.354
MQA	2.929	2.573	2.408	2.304
MLA	2.901	2.564	2.392	2.377
LRKV	**2.893**	**2.509**	**2.376**	**2.288**

- Held-out cross-entropy; lower is better. Compare within columns.
- Relative CE reduction vs. MHA: 0.34%, 0.83%, 0.54%, 1.34%.
- At 6.3B, MQA also beats MHA: stronger sharing need not worsen CE.

Paper, Table 1, PDF p. 5. Relative changes are calculated from the reported CE values.

LRKV reaches baseline quality in fewer training steps



At 2.5B: 18.7% fewer steps to MHA's final loss; 23.6% average across four baselines. The paper's footnote identifies sample efficiency, not a measured per-step or wall-clock speedup.

Original Fig. 3, PDF p. 6; compute-accounting footnote, p. 5. Cache percentages are as reported.

At 2.5B, reasoning gains coexist with a code-generation drop

Task	MHA (%)	LRKV (%)	Difference (pp)
ARC-Easy	66.6	70.7	+4.1
ARC-Challenge	47.1	53.8	+6.7
MMLU	39.3	42.2	+2.9
GSM8K	10.2	11.3	+1.1
HumanEval	13.4	11.7	−1.7

- After the same supervised midtraining mixture.
- Reported five-task average: 35.3% → 37.9% (+2.6 pp).
- The gain is an average advantage, with task-specific trade-offs.

Paper, Table 2, PDF p. 6; midtraining mixture, Appendix D.2, p. 24.

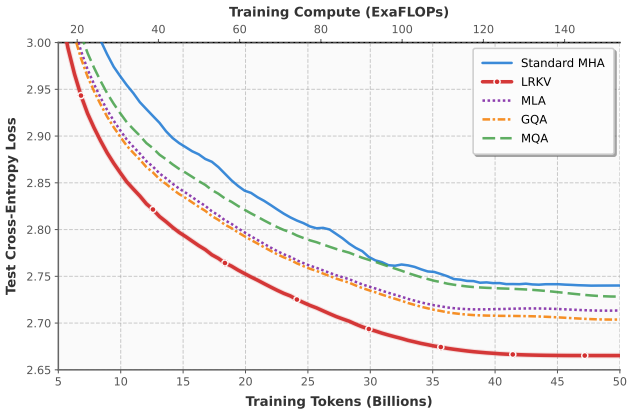
Average downstream gains persist, but exceptions matter

Reported five-task average (%)	128M	2.5B	6.3B
MHA	17.9	35.3	38.8
MQA	17.3	33.3	33.4
MLA	18.0	36.9	36.9
LRKV	**18.9**	**37.9**	**40.2***

- 6.3B HumanEval: LRKV 12.8%, MHA 14.6%.
- 128M GSM8K: LRKV 0.3%, MHA 1.0%.
- Five 2.5B architectures show BPB/score association ($R^2 = 0.786$); this does not identify a causal mechanism.

Table 2, p. 6; Fig. 19, p. 26. *Reported 40.2%; the five listed scores average to 40.36%.

At 8K context, LRKV reports a 2.74% CE reduction



Method	CE ↓
MHA	2.740
GQA	2.704
MQA	2.728
MLA	2.714
LRKV	**2.665**

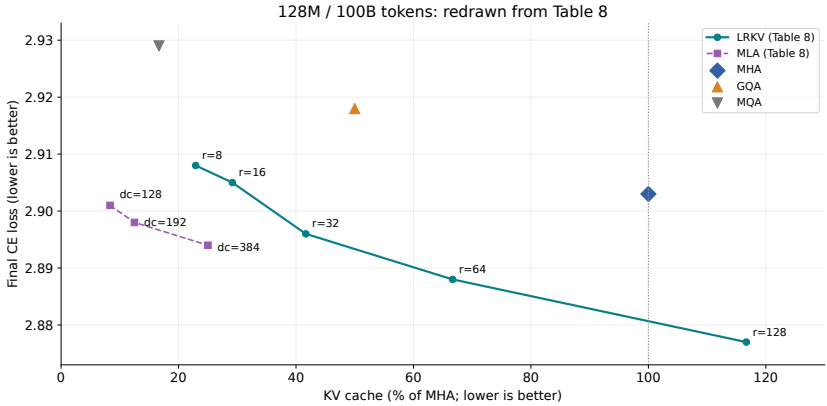
512M label; 50B tokens.

$r = 51$; 48.2% reported cache.

Fixed-context language modeling.

Original Fig. 4, p. 6; precise values from Table 9, p. 26. This does not test context extrapolation.

Table 8 shows a trade-off, not universal dominance over MLA



MLA-384: 25.0% cache, CE 2.894; LRKV-32: 41.7%, CE 2.896.

Redrawn from Table 8, PDF p. 25 (128M, 100B tokens). Original Fig. 5 contains conflicting MLA points.

Head diversity is measured in bilinear-form space

$$A_h = W_h^Q (W_h^K)^\top, \qquad x_i A_h x_j^\top = \text{unscaled linear attention logit.}$$

- Coupled orthogonal changes of Q/K coordinates leave A_h unchanged.
- Form the Gram matrix of normalized A_h; center across heads.

$$G_c = JGJ, \qquad J = I - \frac{\mathbf{1}\mathbf{1}^\top}{H},$$

$$p_i = \frac{\lambda_i(G_c)}{\sum_j \lambda_j(G_c)}, \qquad r_{\text{eff}} = \exp\left(-\sum_i p_i \ln p_i\right).$$

Paper, Sec. 4.3, pp. 7–9; Appendix C.1, p. 15. Static parameter metric; excludes V and input behavior.

LRKV approaches the centered-rank ceiling

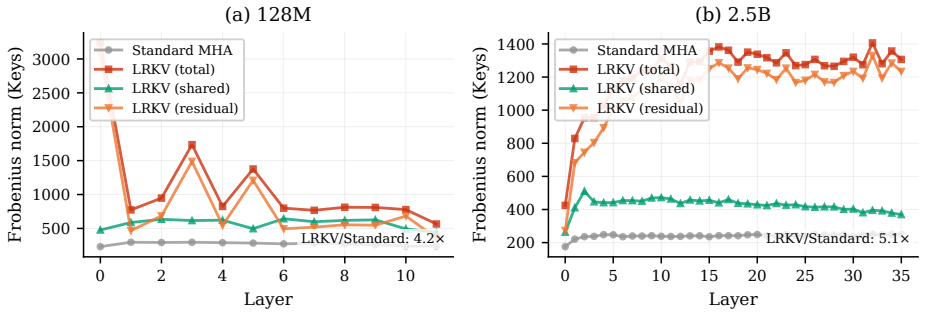
PCA effective rank / H	128M ($H = 6$)	2.5B ($H = 18$)
MHA	82.4%	94.0%
LRKV ($r = 64$)	83.0%	93.5%
MQA	78.4%	91.0%
MLA	78.5%	91.6%
Mathematical ceiling	*83.33%*	*94.44%*

Reader's analysis: $G_c\mathbf{1} = 0$, so $r_{\text{eff}} \leq H - 1$.

- Different head counts change the percentage ceiling.
- MQA retains diversity through independent queries; centering increases its 2.5B percentage from 86.2 to 91.0.

Paper, Table 3, p. 16. Ceiling derived here. The $r = 64$ analysis differs from Table 7's main ranks.

Large weight norms motivate checking normalization placement



- Authors attribute larger LRKV norms to additive optimization and post-projection RMSNorm.
- RMSNorm of a sum is not the sum of separately normalized paths.
- Factored decoding needs consistent per-token, per-head scaling.

Original Fig. 8 and normalization discussion, PDF p. 9. Scaling requirements are reader's analysis.

Full RoPE requires more than linearity

Reader's analysis Use column vectors for the residual key $B_h z_t$.

$$P_t(k_{s,t} + B_h z_t) = P_t k_{s,t} + P_t B_h z_t.$$

To rotate only the low-dimensional latent, we would need

$$\boxed{P_t B_h = B_h \widetilde{P}_t \quad \text{for each position } t.}$$

- P_t rotates d_h dimensions; $\widetilde{P}_t$ would act in r dimensions.
- This requires an invariant residual subspace, absent for arbitrary learned B_h.
- The paper's full-RoPE claim therefore does not follow from Eqs. (6)–(7).

Claim examined: Sec. 3, positional-embedding paragraph, PDF p. 4. A counterexample is in the backup slides.

The measured-cache appendix mixes incompatible numbers

Scale	LRKV / MHA in Table 12	Actual ratio	Printed ratio
128M	48 / 72 MB	66.67%	52.6%
2.5B	360 / 648 MB	55.56%	48.4%
6.3B	612 / 1152 MB	53.13%	45.1%

- Table 12 says $r = 64$; the actual ratios match that rank.
- Main configurations use $r = 46, 51, 55, 54$ instead.
- "Measured" means tensor element counts $\times$ bytes per element.
- Layer counts also conflict: 2.5B is 18 layers in Table 5, 36 in analysis figures.

Paper, Appendix G, pp. 28–29; Table 5, p. 23; Figs. 7–8, pp. 8–9. Ratios calculated here.

The architecture idea is clearer than the deployment evidence

What is useful

A shared dense base plus low-rank head corrections gives an interpretable capacity–cache control. Associativity makes the linear formulation exact without historical K/V expansion.

What the tables report

Better pretraining CE and higher average downstream scores under the reported training recipe, with task-specific regressions.

What a deployment claim still needs

A consistent RoPE/normalization implementation, reconciled configurations and memory figures, and optimized comparisons at matched cache budgets.

Synthesis of Secs. 3–4 and Appendices C, D, F, G. Interpretation by the reader.

References and discussion prompts

James O'Neill, Robert Clancy, Mariia Matskevichus, and Fergal Reid.

Low-Rank Key Value Attention. arXiv:2601.11471v3, 2026.

https://arxiv.org/abs/2601.11471v3

1. How should residual subspaces be constrained to support full RoPE?
2. At equal cache size, when is a shared dense base preferable to a joint latent?
3. Do operator-space diversity measures predict task-specific behavior?

Original figures reproduced from the v3 source, CC BY 4.0. Table 8 plot redrawn with attribution.

Backup: the complete rank sweep clarifies the trade-off

Configuration	Cache / MHA	CE ↓	BPB ↓
LRKV $r = 8$	22.9%	2.908	0.881
LRKV $r = 16$	29.2%	2.905	0.880
LRKV $r = 32$	41.7%	2.896	0.877
LRKV $r = 64$	66.7%	2.888	0.875
LRKV $r = 128$	116.7%	2.877	0.873
MLA $d_c = 128$	8.3%	2.901	0.876
MLA $d_c = 192$	12.5%	2.898	0.876
MLA $d_c = 384$	25.0%	2.894	0.875
MHA	100.0%	2.903	0.878

Paper, Table 8, PDF p. 25. 128M label, 100B tokens, $H = 6$, $d_h = 128$. The printed 12-5% is read as 12.5%.

Backup: a rotation can leave the residual subspace

Reader's analysis Let the residual span only the first coordinate:

$$B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \qquad P = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}, \qquad PB = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

For any scalar c acting on the one-dimensional latent,

$$Bc = \begin{bmatrix} c \\ 0 \end{bmatrix} \neq PB.$$

- Linearity distributes rotation over addition, not through an arbitrary B.
- Pair-aligned or otherwise invariant subspaces could change the result.
- Such constraints would need their own capacity and cache analysis.

Teaching counterexample to the unconditional claim on PDF p. 4; not a model experiment.

Backup: centering itself removes one available dimension

For H perfectly orthogonal normalized operators, $G = I_H$:

$$G_c = JGJ = J, \quad \lambda(G_c) = \underbrace{1, \ldots, 1}_{H-1 \text{ times}}, 0.$$

$$r_{\text{eff}}(G) = H, \qquad r_{\text{eff}}(G_c) = H - 1.$$

- With 18 heads, the percentage falls from 100% to 94.44% even in this ideal case.
- Therefore a roughly five-point decrease alone does not identify a shared mean component.
- Figure 9 is labeled PCA but reaches nearly 6 for six heads; a centered rank cannot exceed 5.

Reader's derivation from Appendix C.1, p. 15. Figure 9 and Table 3 are on p. 16.

Backup: MLA also admits associative computation

With $Z = XW_{\text{down}}$ and $K = ZW_{\text{up}}^{K}$:

$$qK^{\top} = (q(W_{\text{up}}^{K})^{\top})Z^{\top}, \qquad aV = (aZ)W_{\text{up}}^{V}.$$

- This feature bottleneck retains every token position.
- Full historical K/V reconstruction is not necessary in this linear form.
- Appendix F describes association but Table 10 charges MLA for reconstruction.
- Compare optimized kernels under consistent positional-encoding assumptions.

Paper, Appendix F, PDF pp. 27–28. Equations rewritten with dimensionally explicit transposes.

Backup: training and evaluation details affect interpretation

Pretraining

FineWeb-Edu; 32,768 tokens/batch.

Muon + AdamW; cosine decay; 2,000 warmup steps.

Weight decay 0.1; clipping 1.0.

Supervised midtraining

460K SmolTalk + 100K MMLU auxiliary + 8K GSM8K.

One epoch; 524,288 tokens/batch; context 2,048.

Per-task shot counts and code-sampling details are not specified individually. BPB also requires a byte-count denominator; CE/ln 2 alone gives bits per token.

Paper, Appendix D, pp. 23–26; Appendix F.1, p. 27. Metric interpretation by the reader.