Decoupled Bottleneck Attention:

Scaling Efficient Transformers via Low-Rank Semantic Routing

Daniel Owen van Dommelen
Independent Research
theapemachine@gmail.com

December 2025

Abstract

The Key-Value cache in Transformer models scales linearly with sequence length and model dimension, creating a critical memory bottleneck for long-context inference. While techniques like Grouped-Query Attention (GQA) reduce cache size by sharing key-value heads, they preserve the full computational cost of attention scoring in high-dimensional space.

We propose **Decoupled Bottleneck Attention**, an architectural modification that exploits the empirical observation that *semantic routing*—deciding which tokens attend to which—operates in a low-rank subspace ($r \approx 32$), while *positional geometry* requires higher fidelity ($r \approx 64$). By decoupling these concerns into separate projection paths, we achieve:

- **Observed KV-cache savings at our experimental scale (v21/v29).** Reducing the attention interaction dimension yields a consistent $\approx 5.33\times$ FP16 KV-cache reduction in both our v21 toy suite (d_{model}=512, d_{attn}=96) and v29 scaling suite (d_{model}=768, d_{attn}=144); in our v29 implementation this corresponds to a measured $\sim 5.2\times$ FP16 reduction at 128k context. Standard Q4_0 KV-cache quantization composes with this to $\sim$19–21$\times$ end-to-end reduction at the same rank (Section 2.6).
- **Evidence at two scales:** v21 WikiText-2 ablations show that bottlenecking attention can outperform a full-rank baseline; v29 FineWeb-Edu scaling runs show that decoupling (48/96) outperforms a standard baseline, GQA (kv=2), and a matched-rank bottleneck baseline (Table 2)
- **Superior data efficiency compared to Grouped-Query Attention (GQA).** On FineWeb-Edu, our Decoupled Bottleneck model (d_{sem}=48, d_{geo}=96; total $d_{\text{attn}} = 144$) achieves lower validation loss than a GQA baseline with shared KV heads (kv=2) under the same training recipe (mean over three seeds: 5.804 ± 0.087 vs 6.348 ± 0.161), demonstrating that reducing the *interaction rank* is a more effective compression strategy than head sharing.

In our v21 WikiText-2 suite, a simple rank-96 bottleneck *outperforms* the full rank-512 baseline (val loss 5.33 vs 5.37), suggesting that standard Transformers can over-allocate capacity to attention. In our v29 FineWeb-Edu suite, the Decoupled Bottleneck model achieves the best validation loss among baseline, GQA, and a matched-rank Bottleneck baseline (Table 2).

Keywords: Transformer, attention mechanism, low-rank, KV-cache, memory efficiency, quantization, long context, rotary position embeddings

1 Introduction

Modern Transformer architectures [24] achieve remarkable performance across language modeling, translation, and reasoning tasks. However, their quadratic attention complexity and linear KV-cache growth present fundamental scalability challenges for long-context applications.

1.1 The Redundancy Hypothesis

We begin with a simple observation: in a 512-dimensional layer, the neurons are not independent. They move in *sympathetic clusters*—correlated groups that reduce the intrinsic dimensionality of the representation. Prior work on LoRA [12] demonstrated that weight *updates* during fine-tuning are low-rank (typically $r \leq 64$). Recent work on gradient dynamics [20] shows that optimization naturally collapses to low rank. We extend this observation to argue that the *architecture itself*—specifically the attention mechanism—should be structurally constrained to match this intrinsic rank.

Empirical measurements from our experiments show that the *Q/K projection activations* feeding attention have low entropy effective rank (typically in the tens of dimensions, far below the nominal width; Appendix B). This aligns with theoretical analysis by Bhojanapalli et al. [5], who identified a "low-rank bottleneck" in multi-head attention, and recent work by Kobayashi et al. [15] showing that weight decay actively induces rank reduction during training. Wang et al. [26] further demonstrate that attention outputs are approximately 60% low-dimensional, adding to low-dimensional residual subspaces. Crucially, Refael et al. [20] proved that gradient rank *decreases monotonically* during training, asymptotically approaching rank one—providing theoretical justification for why architectural bottlenecks become increasingly appropriate as training progresses. Chiang & Yogatama [6] show that RoPE may cause dimension inefficiency for long-distance retrieval, supporting our use of higher dimensions (64) for the geometric path.

1.2 Comparison with Existing Approaches

Grouped-Query Attention (GQA). While Grouped-Query Attention [2] successfully reduces KV-cache memory by sharing key-value heads across multiple query heads, it maintains the full computational cost of the query projection and attention scoring in the high-dimensional space. Each query still operates in $\mathbb{R}^d$, and every attention score still requires a d-dimensional dot product—GQA merely amortizes the *storage* cost, not the *interaction* cost.

Our Bottleneck approach reduces both memory *and* compute by compressing the interaction manifold. Rather than sharing high-dimensional KV pairs, we project queries and keys into a low-rank semantic subspace ($r \ll d$) *before* computing attention, reducing dot-product complexity from $O(n^2 d)$ to $O(n^2 r)$.

Multi-Head Latent Attention (MLA). DeepSeek-V2 [8] introduced MLA, which compresses KV storage into a latent vector, achieving 93% cache reduction. However, MLA *up-projects* during the forward pass to perform attention in the original high-dimensional space. Our method remains low-rank throughout, saving both memory and compute.

Disentangled Attention. DeBERTa [10] pioneered the separation of content and position representations in attention scoring. We adopt this disentanglement principle but leverage it for *efficiency*: applying aggressive compression to the semantic (content) path while preserving fidelity in the geometric (position) path.

1.3 Contributions

1. We demonstrate that attention routing can be performed in $\sim$32 dimensions without perplexity degradation, while positional encoding requires $\sim$64 dimensions for RoPE fidelity.

2. We propose **Decoupled Bottleneck Attention**, which separates semantic and geometric scoring paths with asymmetric dimensionality.

3. We introduce and **evaluate** a **Null Token** mechanism that provides an explicit "attend nowhere" option (beneficial in some low-rank regimes, but not universally; Appendix D).

4. We report measured KV-cache footprints for our implementations and show that reducing the interaction dimension yields a consistent $\approx 5.33\times$ FP16 KV-cache reduction at our experimental scales (v21/v29). We discuss how this composes with standard KV-cache quantization; large fixed-rank scaling numbers are treated as non-validated arithmetic, not a central claim (Section 2.6).

2 Related Work

Low-rank and approximate attention. A long line of work seeks to reduce the quadratic cost of attention by approximating the score computation or constraining its rank. Linformer [25] projects keys and values into a lower-dimensional subspace along the sequence dimension, yielding linear-time attention under a low-rank assumption. Kernel and hashing methods such as Performer [7] and Reformer [14] similarly reduce attention cost via randomized features or locality-sensitive hashing. Our setting is different: we train standard causal LMs, but explicitly reduce the query/key interaction dimension inside each layer, targeting both compute ($O(n^2 r)$) and KV-cache memory ($O(nr)$).

Sparse/local attention for long documents. Sparse patterns (e.g., sliding window with global tokens) as in Longformer [4] and BigBird [27] reduce attention compute while retaining access to distant context. However, for autoregressive decoding these methods still accumulate a KV cache whose size grows linearly with context length. Our work instead reduces the per-token cache footprint, which is complementary to sparse attention and other long-context strategies [13].

KV-cache optimization. Sharing KV heads reduces cache storage by amortizing keys and values across query heads (MQA/GQA) [21, 2]. Latent KV schemes such as MLA compress the cache into a lower-dimensional latent that is expanded during attention [8]. Orthogonally, quantizing the KV cache reduces memory at fixed architecture [11, 17]. Our decoupled bottleneck reduces the interaction dimension before scoring (saving compute) and also makes heterogeneous KV quantization natural: semantic keys can often be quantized more aggressively than geometric keys.

Expressiveness limits and structured alternatives. Reducing interaction rank too far can harm representation power: theory and empirical evidence show regimes where increasing heads under fixed head dimension does not recover lost capacity [5, 3]. Recent structured-matrix formulations aim to increase effective rank without full cost by parameterizing attention maps with richer structured operators [16]. Decoupling is a simple architectural compromise: we keep a higher-dimensional geometric path (with RoPE) while aggressively compressing only the semantic routing path.

3 Methodology

3.1 Standard Multi-Head Attention

In standard scaled dot-product attention with H heads:

$$\text{Attn}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V \tag{1}$$

where $Q, K, V \in \mathbb{R}^{n \times d}$ are obtained by linear projection from the input $X \in \mathbb{R}^{n \times d_{\text{model}}}$:

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V \tag{2}$$

with $W_Q, W_K, W_V \in \mathbb{R}^{d_{\text{model}} \times d}$. For language modeling with context length n and dimension d, the KV-cache requires $O(2 \cdot L \cdot n \cdot d)$ memory, where L is the number of layers.

3.2 Bottleneck Attention

We introduce a simple modification: project Q and K to a lower-dimensional space *before* computing attention scores.[1]

$$Q' = XW'_Q, \quad K' = XW'_K \tag{3}$$

where $W'_Q, W'_K \in \mathbb{R}^{d_{\text{model}} \times d_{\text{attn}}}$ with $d_{\text{attn}} \ll d_{\text{model}}$. The attention computation becomes:

$$\text{Attn}_{\text{bottleneck}}(Q', K', V') = \text{softmax}\left(\frac{Q'K'^{\top}}{\sqrt{d_{\text{attn}}/H}}\right)V' \tag{4}$$

This reduces the dot-product complexity from $O(n^2 \cdot d_{\text{model}})$ to $O(n^2 \cdot d_{\text{attn}})$ and the KV-cache from $O(n \cdot d_{\text{model}})$ to $O(n \cdot d_{\text{attn}})$.

3.3 Decoupled Bottleneck Attention

The key insight motivating decoupling is that *semantic matching* ("is this token semantically related?") and *geometric positioning* ("how far away is this token?") have different intrinsic dimensionality requirements.

We decompose the attention score into two additive components:

$$\text{Score} = \underbrace{\frac{Q_{\text{sem}}K_{\text{sem}}^{\top}}{\sqrt{d_{\text{sem}}/H}}}_{\text{Semantic Path}} + \underbrace{\frac{Q_{\text{geo}}K_{\text{geo}}^{\top}}{\sqrt{d_{\text{geo}}/H}}}_{\text{Geometric Path}} \tag{5}$$

where:

$$Q_{\text{sem}} = XW_{Q,\text{sem}}, \quad K_{\text{sem}} = XW_{K,\text{sem}} \qquad (d_{\text{sem}} = 32) \tag{6}$$

$$Q_{\text{geo}} = XW_{Q,\text{geo}}, \quad K_{\text{geo}} = XW_{K,\text{geo}} \qquad (d_{\text{geo}} = 64) \tag{7}$$

Critically, we apply **Rotary Position Embeddings (RoPE)** [22] *only* to the geometric path:

$$Q_{\text{geo}}, K_{\text{geo}} \leftarrow \text{RoPE}(Q_{\text{geo}}, K_{\text{geo}}, \text{position}) \tag{8}$$

The semantic path operates on pure content similarity, while the geometric path encodes positional relationships. The value projection uses the combined dimension:

$$V = XW_V, \quad W_V \in \mathbb{R}^{d_{\text{model}} \times d_{\text{attn}}} \tag{9}$$

where $d_{\text{attn}} = d_{\text{sem}} + d_{\text{geo}}$. In our smaller v21 suite we use a 32/64 split (d_{attn}=96), and in our v29 scaling suite we use a 48/96 split (d_{attn}=144).

[1] Our use of "bottleneck" refers to dimensionality reduction in the query/key space, distinct from Park et al.'s BAM [19], which applies channel and spatial attention in CNNs for computer vision.

3.4 The Null Token Mechanism

Low-rank attention can become unstable when queries lack semantically appropriate keys. We introduce a learnable **null token** $k_\emptyset$ providing an explicit "attend nowhere" option:

$$\text{Score}_{\text{null}} = \frac{Q_{\text{sem}} k_{\emptyset,\text{sem}}^\top}{\sqrt{d_{\text{sem}}/H}} + \frac{Q_{\text{geo}} k_{\emptyset,\text{geo}}^\top}{\sqrt{d_{\text{geo}}/H}} \tag{10}$$

This score is concatenated to the attention matrix before softmax, allowing the model to "dump" attention mass when no key is appropriate, which can stabilize training at very low ranks. In our v29 48/96 setting, however, the null token is not beneficial under this recipe (Appendix D).

3.5 Tied Q-K Projections

For the semantic path, we optionally **tie** the query and key projections: $W_{Q,\text{sem}} = W_{K,\text{sem}}$. This enforces symmetric similarity ("A attends to B iff B attends to A"), which is appropriate for content matching but not for position-dependent relationships.

3.6 Quantized Inference

For inference, we apply aggressive quantization to the KV-cache. Recent work has demonstrated that 4-bit KV cache quantization preserves model quality remarkably well. Turboderp's ExLlamaV2 implementation [23] showed Q4 cache performs comparably to FP16, and this capability has been integrated into production inference engines like llama.cpp [9]. We implement block-wise Q4_0 quantization following this approach:

$$x_{\text{quantized}} = \text{round}\left(\frac{x}{\text{scale}}\right), \quad \text{scale} = \frac{\max(|x_{\text{block}}|)}{7} \tag{11}$$

where each block of 32 elements shares a single FP16 scale factor. In the idealized limit (ignoring scale metadata) 4-bit values correspond to 0.5 bytes/value. With Q4_0 block scales, the effective bytes/value is slightly larger (18 bytes per 32 values $\Rightarrow$ 0.5625 bytes/value), so the ideal 4× factor becomes $\approx 3.56\times$ in practice. Combined with dimension reduction, the per-layer KV-cache reduction is approximately:

$$\text{Compression} \approx \underbrace{\frac{d_{\text{model}}}{d_{\text{attn}}}}_{\text{Dimension}} \times \underbrace{\frac{2 \text{ bytes}}{0.5 \text{ bytes}}}_{\text{Ideal Q4}} \approx \frac{d_{\text{model}}}{d_{\text{attn}}} \times 4. \tag{12}$$

Both our v21 (512→96) and v29 (768→144) settings yield the same dimension factor (5.33×), implying ∼19–21× end-to-end FP16→Q4 savings depending on quantization overheads.

Scaling arithmetic (context only; not validated at scale). The KV-cache memory at long context depends on the choice of attention dimension d_{attn} at scale. For a rough Llama-like configuration (32 layers, $d_{\text{model}} = 4096$, 128k context, batch=1), the FP16 KV cache is:

$$M_{\text{FP16}} \approx 2 \cdot 32 \cdot 4096 \cdot 128\text{k} \cdot 2 \text{ bytes} \approx 64 \text{ GiB}.$$

With 4-bit KV-cache quantization (idealized 0.5 bytes/value), the memory becomes:

$$M_{\text{Q4}} \approx 2 \cdot 32 \cdot d_{\text{attn}} \cdot 128\text{k} \cdot 0.5 \text{ bytes}.$$

This yields two reference scenarios (for context):

- **Constant-fraction d_{attn} (e.g., $d_{\text{attn}} = 768$):** $M_{\text{Q4}} \approx 3.0$ GiB, for an overall reduction of $\sim 21\times$.

- **Speculative fixed-rank d_{attn} (intuition only):** If one could keep d_{attn} roughly constant while scaling d_{model} (e.g., d_{attn}=96 at d_{model}=4096), the same linear arithmetic yields an $\mathcal{O}(10^2)$ reduction versus a standard FP16 baseline.[2]

The architectural contribution is the *dimension reduction* (the ratio $4096/d_{\text{attn}}$); the additional factor of 4× comes from standard 16→4-bit quantization (idealized). For fair comparisons, note that GQA caches can also be quantized; e.g., an 8× GQA KV cache with Q4 would already yield ∼32× reduction vs FP16 standard. We therefore treat fixed-rank scaling numbers as back-of-the-envelope upper bounds, not a primary experimental claim.

Heterogeneous KV-cache quantization (decoupled). A practical benefit of decoupling is that it enables *heterogeneous* KV-cache quantization: we can compress the semantic path more aggressively (e.g., Q4) while keeping the geometric (RoPE) path at higher fidelity (e.g., Q8). As a lightweight sanity check, we compared FP16 caches to a decoupled policy with K_{sem} = Q4, K_{geo} = Q8, V = Q4 and a 128-token FP16 residual window. On a small held-out calibration slice from FineWeb-Edu, this policy yielded $\Delta\text{NLL} \approx 0.015$ nats/token (perplexity ratio ≈ 1.015) and $\text{KL}(p_{\text{FP16}} \| p_{\text{quant}}) \approx 0.006$ nats/token, while preserving identical greedy decoding for short prompts. These guardrail metrics suggest that aggressive semantic compression can be applied without catastrophic degradation, even though stochastic sampling trajectories may diverge due to small distribution shifts.

While we report training throughput in our experiments, the theoretical FLOPs reduction in the attention mechanism ($O(n^2 d) \to O(n^2 r)$) implies a proportional speedup in the *prefill phase* of inference, where the KV-cache is populated. For autoregressive decoding, the memory bandwidth savings from the smaller cache dominate latency improvements.

4 Experiments

4.1 Experimental Setup

Why two suites? We use a two-stage experimental strategy: the **v21 ablation suite** (WikiText-2, smaller model) enables rapid iteration and mechanistic insight, while the **v29 scaling suite** (FineWeb-Edu, larger model) tests whether the same architectural ideas hold at higher scale. Because these suites differ in model size, data distribution, and training details, we report them separately and avoid direct numerical comparison across suites.

Model Configuration. We report results from two experiment suites:

- **v21 (ablation suite):** a smaller 6-layer model (d_{model}=512) trained on WikiText-2 for rapid architectural iteration.
- **v29 (scaling suite):** a larger 12-layer model (d_{model}=768, 12 heads, d_{ff}=3072) trained on FineWeb-Edu (100M-token dataset) to validate scaling behavior.

Datasets.

- **WikiText-2**: 2M tokens of Wikipedia text used for rapid prototyping in the v21 ablation suite.
- **FineWeb-Edu**: a 100M-token dataset of educational web content used for the v29 scaling suite. In our implementation, we use a 50,257-token vocabulary (GPT-2 compatible) for tokenized data.

[2]This is the origin of the often-quoted "168×": $(4096/96) \times 4 \approx 171$, sometimes rounded. Including Q4_0 scale metadata gives $(4096/96) \times (2/0.5625) \approx 152$. We do *not* validate fixed-rank scaling in this work.

Training. Unless otherwise noted, v29 models are trained for 6000 steps with:

- Lion optimizer (β_1=0.9, β_2=0.99), constant learning rate 3×10^{-4}, weight decay 0.1, warmup 0
- Batch size 8 with gradient accumulation 2 (global batch = 16 sequences/update), gradient clipping at 1.0
- Training context length: 256 tokens (block size 256); RoPE extrapolation is evaluated separately at 2× and 4× this length (Appendix C)
- BF16 parameters with AMP (BF16), dropout 0.0, SwiGLU MLP

All v29 runs in Table 2 were executed with the same recipe on Apple Silicon via MPS; full configurations are recorded in the run logs.

4.2 v21 Ablation Suite: WikiText-2 Results

Table 1: WikiText-2 Validation Loss Comparison (v21 suite; 6-layer model)

Model	Attn Config	Params	Val Loss	Tok/s
Standard Baseline	$d = 512$	31.8M	5.37	20k
Combined 96	$d_{\text{attn}} = 96$	30.1M	**5.33**	117k
Bottleneck 128	$d_{\text{attn}} = 128$	31.3M	5.48	128k
Decoupled 32/64	d_{sem}=32, d_{geo}=64	30.9M	5.59	106k
GQA (kv=2)	8Q/2KV heads	30.1M	5.63	25k
Small Model	$d_{\text{model}} = 128$	4.2M	5.74	930k

Key Finding (v21). Table 1 shows that the Combined 96 bottleneck achieves the *lowest* validation loss (5.33), outperforming the full-rank baseline (5.37). This supports the hypothesis that attention routing can be learned in a substantially lower-dimensional interaction space.

4.3 v29 Scaling Suite: FineWeb-Edu Results

To validate that our findings generalize beyond small datasets, we train on a 100M-token FineWeb-Edu dataset (Figure 2).

Table 2: FineWeb-Edu Validation Loss (100M-token dataset; 256 context; 6000 steps; v29 suite; mean ± std over three seeds)

Model	Attn Config	Params	Val Loss	Val PPL
Standard Baseline	$d_{\text{attn}} = d_{\text{model}} = 768$	139.8M	6.385 ± 0.222	602.9 ± 140.4
GQA (kv=2)	12Q/2KV	128.0M	6.348 ± 0.161	576.5 ± 95.5
Bottleneck (rank 144)	$d_{\text{attn}} = 144$	116.8M	6.424 ± 0.090	617.9 ± 56.9
Decoupled 48/96	d_{sem}=48, d_{geo}=96	**116.8M**	$\mathbf{5.804 \pm 0.087}$	$\mathbf{332.3 \pm 28.5}$

Scaling Observation. On FineWeb-Edu (100M-token dataset), the Decoupled Bottleneck model *outperformed* the Standard Baseline, a reduced-KV GQA baseline (kv=2), and a matched-rank Bottleneck baseline (Table 2). We observe substantial run-to-run variability for the full-rank baseline under this short training regime; therefore, we report mean ± std over three seeds in Table 2, and show representative learning curves for a single seed in Figure 2. To address baseline tuning concerns, we include a small baseline learning-rate sweep and a longer-horizon comparison (seed 1337) in Appendix E.

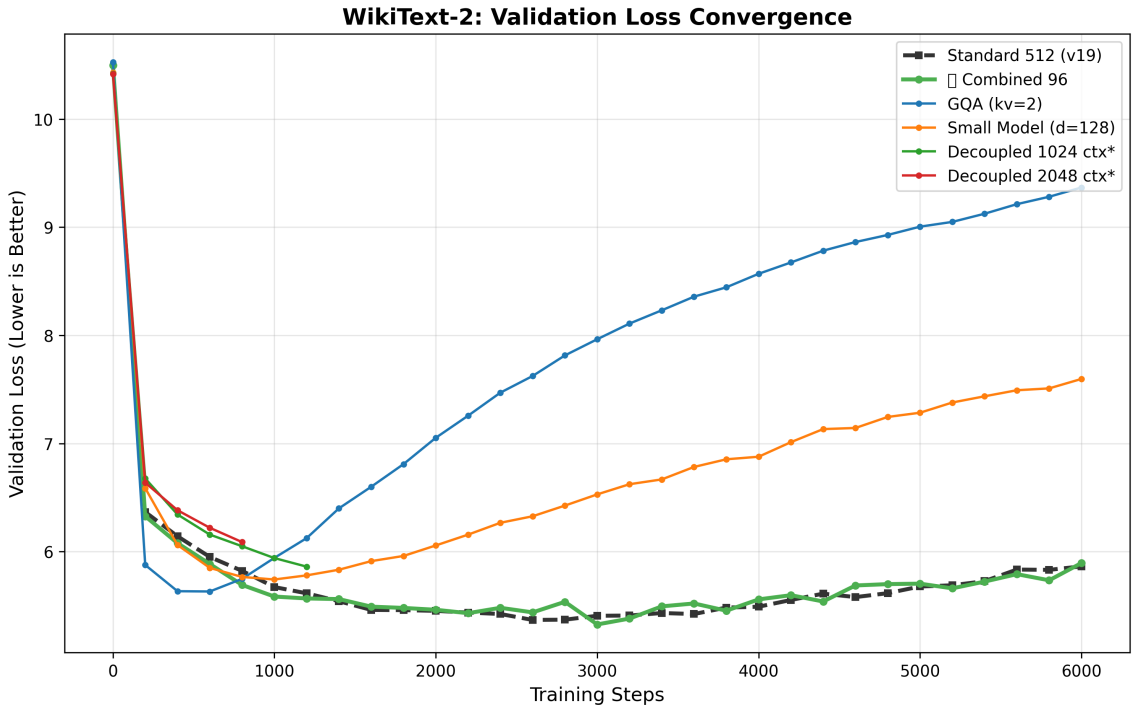


Figure 1: Validation loss curves on WikiText-2 (v21 suite). Bottlenecked models converge rapidly; Combined 96 achieves the best final loss.

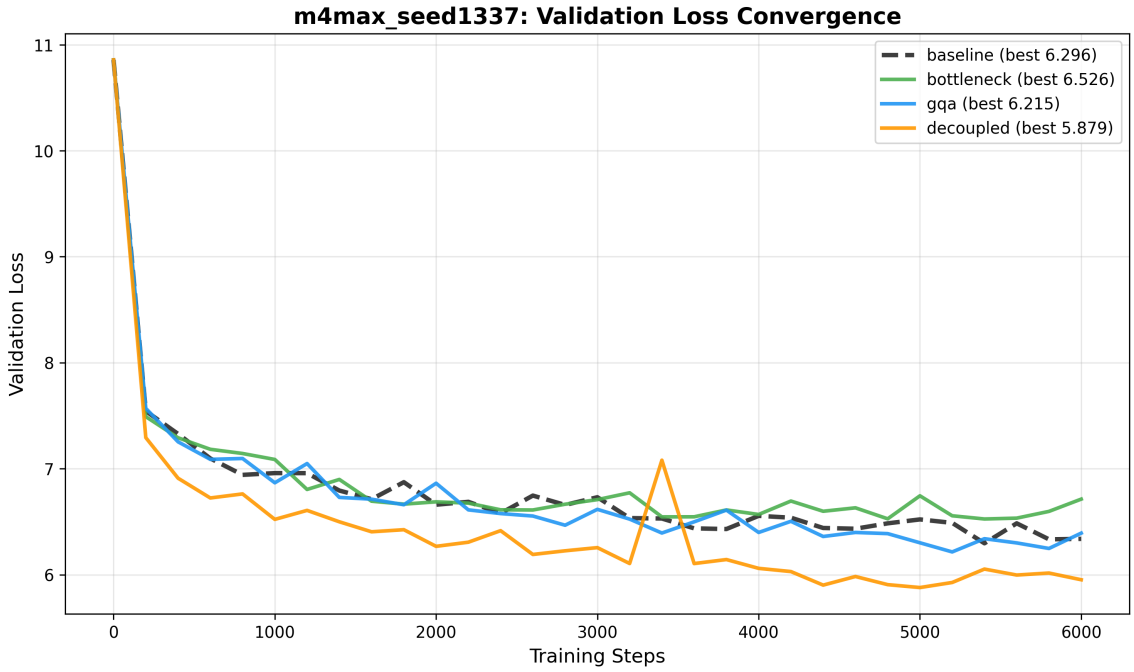


Figure 2: Validation loss curves on FineWeb-Edu (100M-token dataset; seed 1337). Under the same training setup, the Decoupled Bottleneck model converges to the lowest validation loss, outperforming both the standard baseline and GQA.

4.4 Final Loss Comparison

Figure 3 provides a direct comparison of final validation losses across our FineWeb-Edu (v29) architectures. The Decoupled Bottleneck (48/96) achieves the best validation loss among baseline, GQA, and bottleneck variants.

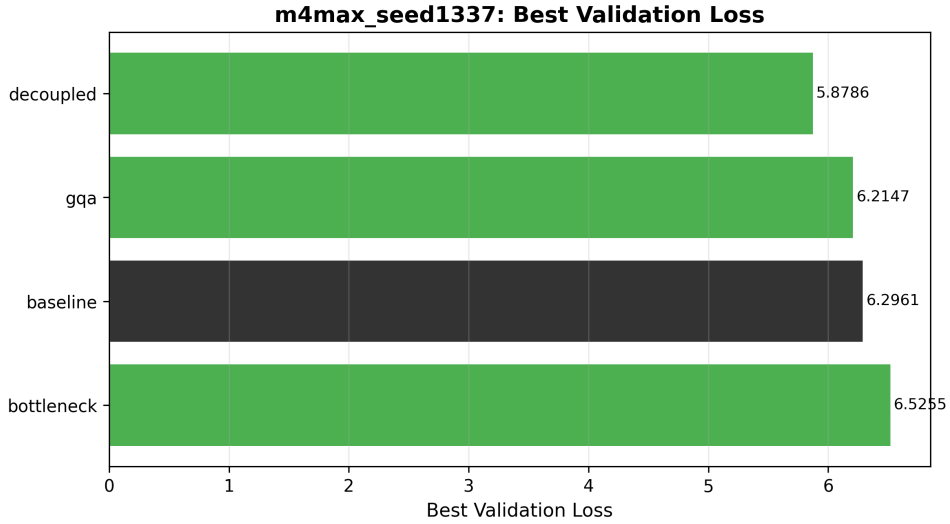


Figure 3: Final validation loss comparison (FineWeb-Edu, v29; seed 1337). Lower is better. Decoupled 48/96 achieves the best loss, outperforming both the standard baseline and a GQA baseline (kv=2).

4.5 Ablation Studies

Wide Residual Stream Hypothesis. Comparing "Small Model" ($d_{\text{model}} = 128$) to "Bottleneck 128" ($d_{\text{model}} = 512$, $d_{\text{attn}} = 128$), we observe a 0.26 loss gap (5.74 vs 5.48) and severe overfitting in the small model. This confirms that the *residual stream* must remain wide; only the *attention interaction* can be compressed.

GQA vs. Bottleneck (Head Sharing vs. Interaction Rank). We compared our method against Grouped-Query Attention (GQA) with matched KV memory footprints on FineWeb-Edu. While GQA reduces KV cache storage by sharing key-value heads, it retains full-rank query projections and attention scoring in high-dimensional space. In contrast, the Bottleneck architecture reduces the interaction dimension directly. In our v29 runs, both GQA (kv=2) and Bottleneck (rank 144) reduce KV-cache memory to ~ 0.75 GB and ~ 0.84 GB at 128k context respectively, but Bottleneck did not consistently improve validation loss over GQA under this recipe (Table 2).

Decoupled vs. Bottleneck (Separating Content and Geometry). Holding the same total attention dimension fixed ($d_{\text{attn}} = 144$), the Decoupled Bottleneck model (48/96) improves validation loss at essentially the same KV cache footprint (~ 0.84 GB at 128k context), suggesting that semantic/geometric separation can improve optimization and generalization at this scale (Table 2).

Long Context Stability. Appendix C reports two lightweight probes: teacher-forced RoPE extrapolation (256/512/1024 contexts) and a passkey needle-in-a-haystack prompt. These probes are intended as regression tests (not downstream evaluations); we report likelihood-based needle effects because strict next-token retrieval accuracy is often 0% for base LMs.

4.6 Memory-Quality Trade-off

Figure 4 shows the quality–efficiency trade-off for our FineWeb-Edu (v29) runs. For these experiments, KV-cache memory at 128k context is measured directly from the implementation, and we compare it against the best validation loss achieved by each architecture. Notably, Decoupled 48/96 matches the Bottleneck's KV footprint (same total d_{attn}) while improving quality, and it outperforms GQA despite GQA's smaller KV cache.

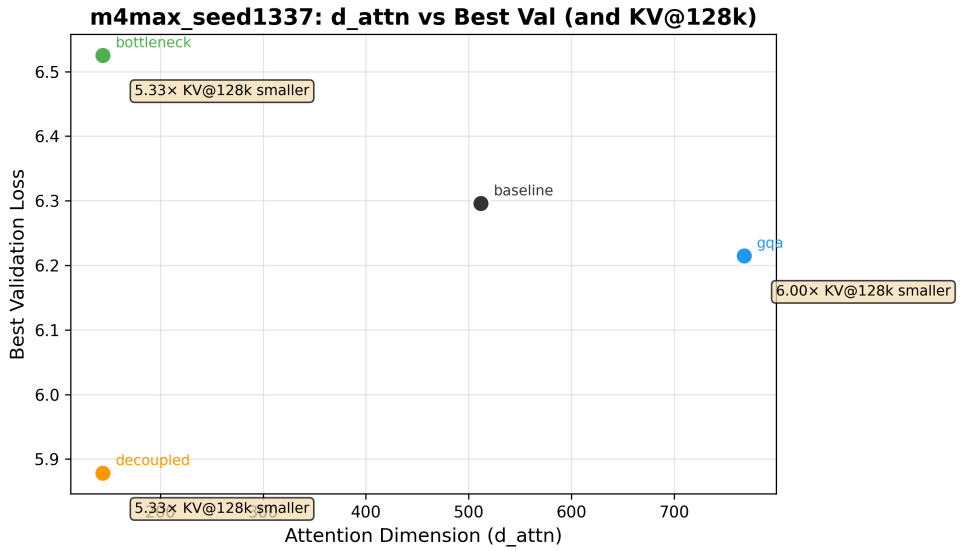


Figure 4: Quality–efficiency comparison on FineWeb-Edu (v29). Points show best validation loss versus attention dimension (d_{attn}), with annotations derived from measured KV-cache memory at 128k context.

4.7 Memory Footprint Analysis

Table 3 gives an illustrative KV-cache scaling projection for a 128k context in a Llama-like configuration (32 layers, $d_{\text{model}} = 4096$). We intentionally *do not* foreground optimistic fixed-rank "upper bound" numbers here; the experimentally grounded takeaway is the linear dependence on d_{attn} and the $\approx 5.33\times$ FP16 reduction that follows from our toy-scale choices ($512 \to 96$ in v21; $768 \to 144$ in v29), which we measure directly in our v29 implementation.

Table 3: KV-Cache Memory for 128k Context (Llama-like scale; projected)

Architecture	**VRAM**	**Compression**
Standard (FP16)	64.0 GB	1×
GQA (32Q/4KV; FP16)	8.0 GB	8×
GQA (32Q/4KV; Q4, ideal)	2.0 GB	32×
MLA (FP16)	4.3 GB	15×
Bottleneck (FP16)	1.5 GB	43×
Decoupled (Q4, constant-fraction d_{attn}=768)	3.0 GB	21×

5 Discussion

5.1 Why Does Low-Rank Attention Work?

We hypothesize two complementary explanations:

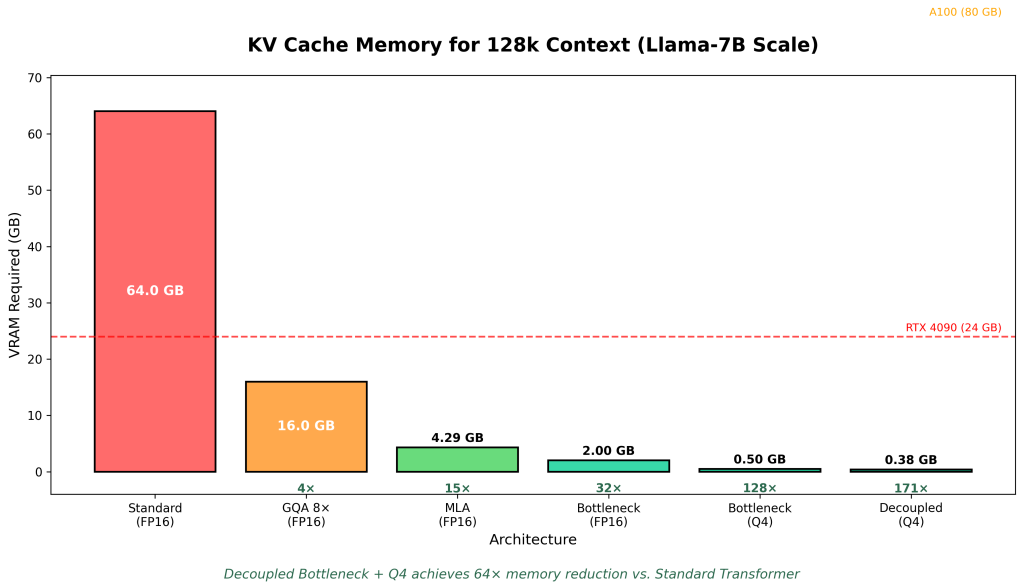


Figure 5: KV-cache memory comparison at 128k context (illustrative scaling projection). The experimentally grounded point is that KV-cache memory scales linearly with d_{attn}; combining this with standard KV-cache quantization yields multiplicative savings. Large fixed-rank upper-bound numbers are omitted from the main narrative and treated as speculative arithmetic (Section 2.6).

Intrinsic Dimensionality. Following Aghajanyan et al. [1], natural language representations lie on low-dimensional manifolds. The attention mechanism's role is *routing*—selecting which tokens to aggregate—not computing complex transformations. Routing decisions are inherently low-entropy and thus low-rank.

Regularization Effect. The bottleneck acts as an implicit regularizer, preventing the model from memorizing spurious token-pair correlations. In our v29 FineWeb runs, reducing the interaction rank improves generalization relative to the full-rank baseline.

Gradient Rank Dynamics. AdaRankGrad [20] proves that gradient rank decreases monotonically during training, eventually approaching rank one. This suggests that *architectural* bottlenecks become increasingly appropriate as training progresses—the model naturally "wants" to operate in a low-rank subspace. By hard-wiring this constraint from the start, we may accelerate convergence by matching the architecture to the optimization landscape.

5.2 When to Use Each Architecture

Our experiments reveal a two-stage story: v21 ablations on WikiText-2 are useful for rapid iteration and mechanistic insight, while v29 FineWeb-Edu results validate the key claims at larger scale.

- **Decoupled Bottleneck:** On FineWeb-Edu (v29), Decoupled 48/96 achieves the best validation loss among the tested variants (baseline, GQA, and bottleneck) while preserving the KV memory benefits of low-rank attention. It is also the *only* architecture enabling **heterogeneous quantization**—e.g., Q4 for semantic and Q8 for geometric paths—which is critical for long-context inference deployments.
- **Standard Attention:** A strong baseline and simplest implementation, but can be memory-inefficient for long contexts.

Recommendation. For *training*, use the v21 ablation suite to quickly explore attention-rank and decoupling choices, then validate at scale with the v29 FineWeb suite. For *inference* under memory constraints, convert to Decoupled with heterogeneous quantization (aggressively compress semantic, preserve geometric fidelity).

Flash/SDPA compatibility. Decoupled Bottleneck Attention can be implemented using PyTorch's fused `scaled_dot_product_attention` by concatenating the scaled semantic and geometric Q/K projections along the head dimension, making it compatible with modern Flash Attention kernels.

5.3 Limitations

- **Sensitivity to setup:** Results can depend on the training recipe and implementation details. Earlier (v21) FineWeb runs showed a degradation for decoupled attention, whereas the v29 decoupled model improves substantially. This motivates replication across seeds, longer training, and additional datasets.
- Experiments are limited to ∼140M parameter models (up to d_{model}=768) trained on a 100M-token dataset. Verification at 7B+ scale is needed.
- The optimal $(d_{\text{sem}}, d_{\text{geo}})$ split may vary with model scale.
- We have not evaluated on downstream tasks (e.g., MMLU, HellaSwag).
- Throughput measurements are from training; inference latency benchmarks are future work.

6 Conclusion

We have demonstrated that attention in Transformers contains significant redundancy. On FineWeb-Edu (100M-token dataset), the Decoupled Bottleneck architecture **surpassed** the standard baseline, a GQA baseline (kv=2), and a matched-rank Bottleneck baseline under the same training recipe (Table 2). These results suggest that, at this scale and training horizon, standard high-dimensional attention can be over-parameterized; larger-scale validation and longer training are important future work.

The core insight is architectural: **Attention is a router, not a processor.** The heavy computation should happen in the feedforward layers (which we leave at full rank), while attention merely selects which tokens to aggregate. By matching the architecture to this functional role, we unlock dramatic efficiency gains.

Our Decoupled Bottleneck Attention separates semantic matching from positional geometry, allowing aggressive compression on the former while preserving RoPE fidelity on the latter. Combined with 4-bit KV-cache quantization, the memory arithmetic suggests that 128k-context inference can become *feasible* on consumer hardware under fixed-rank scaling assumptions (Figure 5); however, this is a projection and we do not claim validated 128k *quality* in this work.

Future Work. We plan to: (1) validate at 7B+ scale where the efficiency gains compound; (2) explore learned mixing weights between semantic and geometric paths; (3) test robustness across seeds, longer training, and additional datasets to understand when decoupling improves optimization; and (4) benchmark inference latency on production hardware. In parallel, we are migrating from the monolithic research prototype to a modular production-oriented implementation (see the repository's `production/` code), with an evolving self-optimization layer for KV-cache policies and resource-aware inference; future work will validate functional parity and quantify real-world latency/memory trade-offs.

Statements and Declarations

Conflict of Interest. The author declares no competing interests. This research was conducted independently without corporate affiliation or funding from entities with financial interests in the outcomes.

Data Availability. All datasets used in this study are publicly available: WikiText-2 [18] is available from Salesforce Research, and FineWeb-Edu is available from Hugging Face. The code, trained model checkpoints, and all experimental logs are available at `https://github.com/theapemachine/experiments`. For convenience and long-term artifact access, we also provide a mirror containing the original run directories (including all checkpoints) at `https://drive.google.com/drive/folders/1sQYMijP06a4Mf_1sEj_gLkE8OOEfUCB6?usp=drive_link`.

Funding. This research was conducted without external funding. All computational resources were provided by the author.

A Replication Plots for FineWeb-Edu (v29; Seeds 1338 and 1339)

Table 2 reports mean ± std over three seeds. In the main text, we show representative learning curves and summary plots for seed 1337; here we include the corresponding plots for seeds 1338 and 1339 to make the replication evidence explicit.

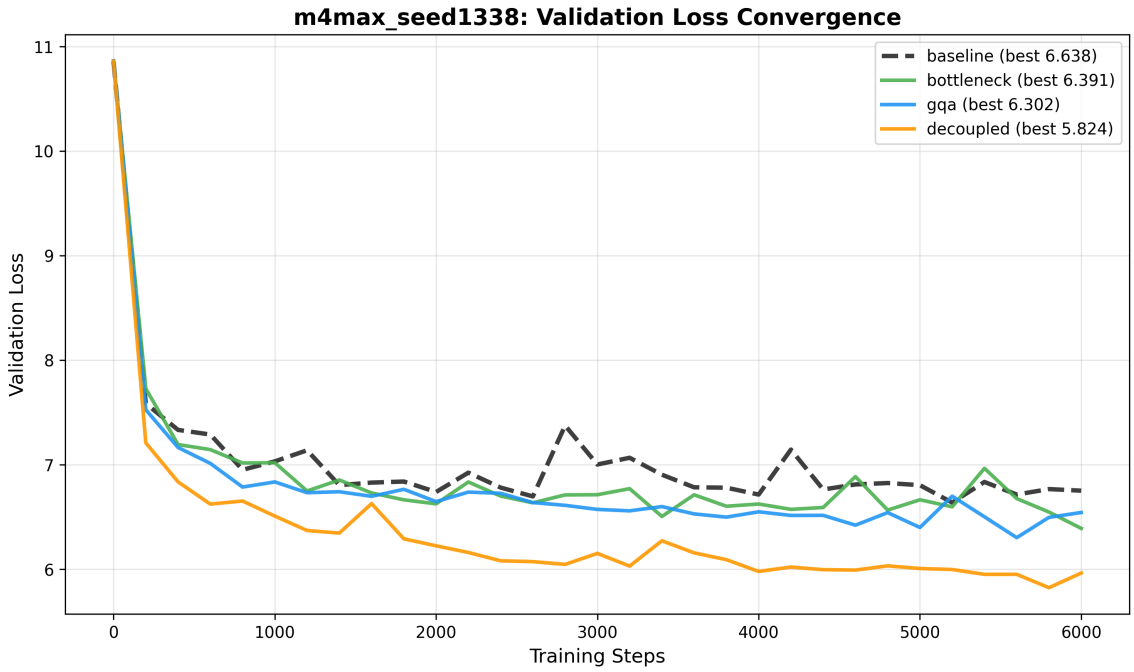


Figure 6: Validation loss curves on FineWeb-Edu (100M-token dataset; seed 1338).

B Effective Rank Evidence (Q/K Projection Activations)

We compute singular value spectra and entropy-based effective rank for the *Q/K projection activations* (post-linear outputs feeding attention) in trained checkpoints, by capturing projection outputs on a fixed token slice and computing singular values of their empirical covariance. We

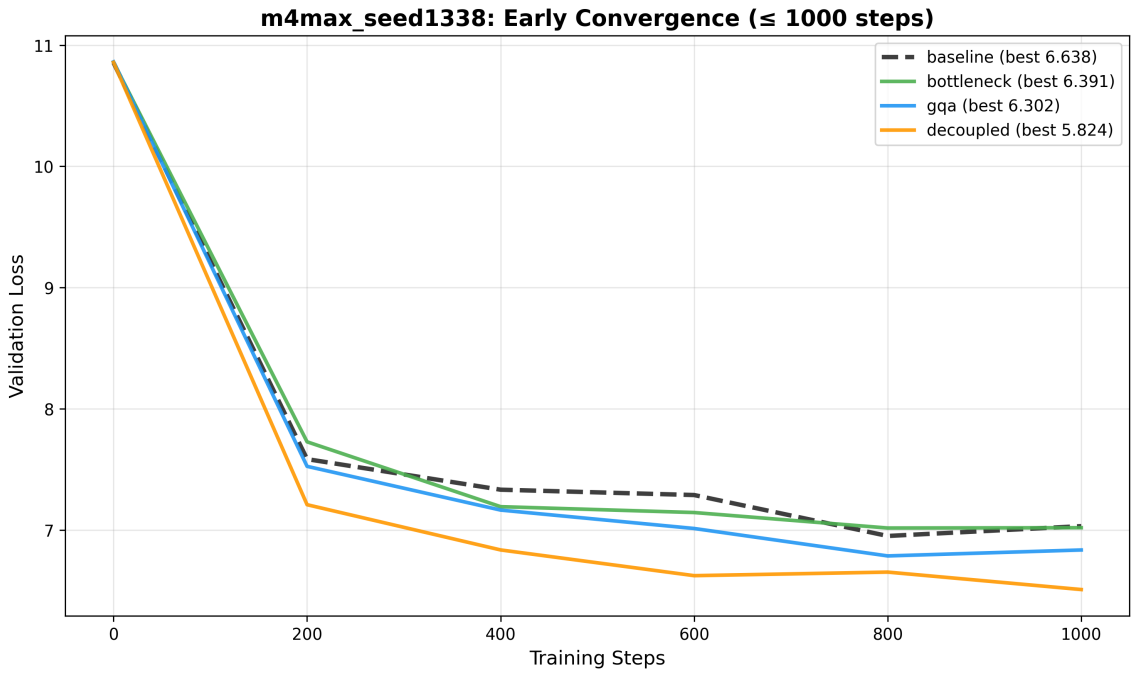


Figure 7: Early convergence view on FineWeb-Edu (seed 1338), highlighting the first phase of optimization.

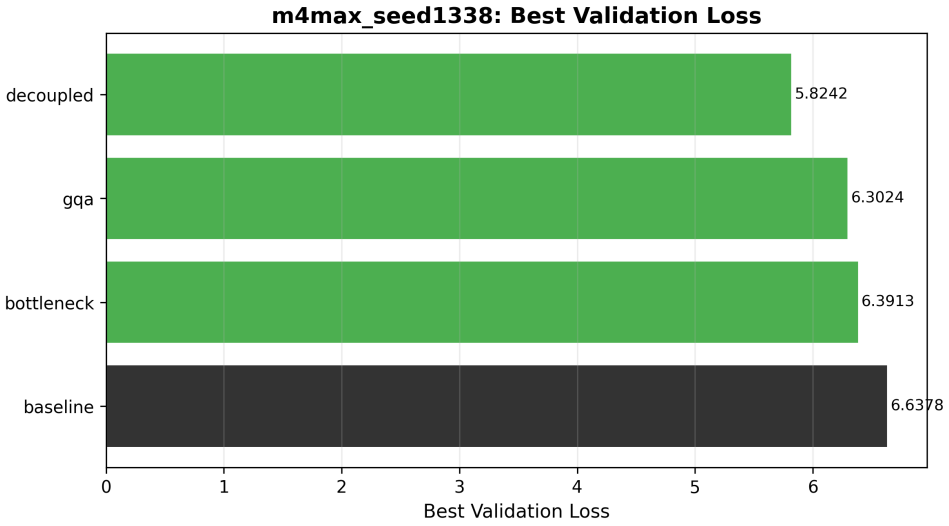


Figure 8: Final validation loss comparison on FineWeb-Edu (v29; seed 1338). Lower is better.

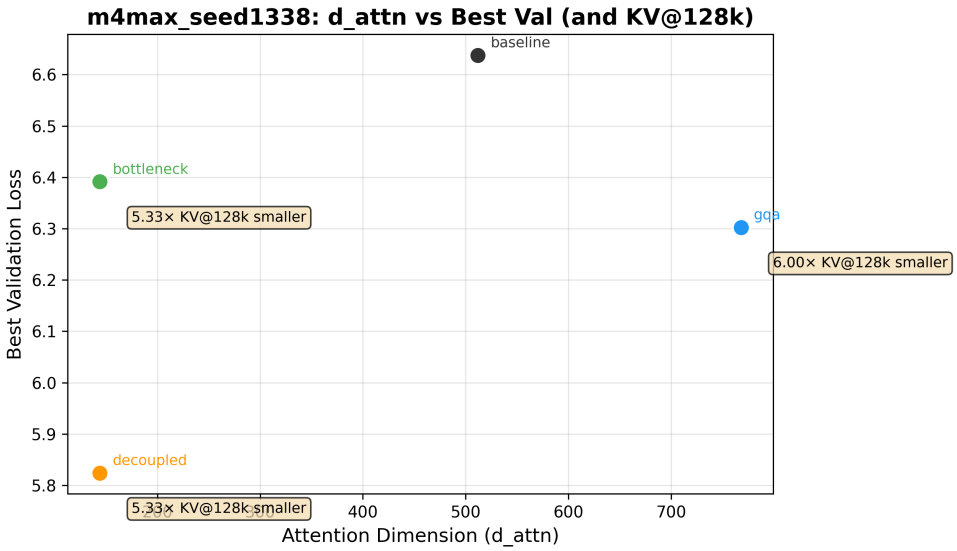


Figure 9: Quality–efficiency comparison on FineWeb-Edu (v29; seed 1338).

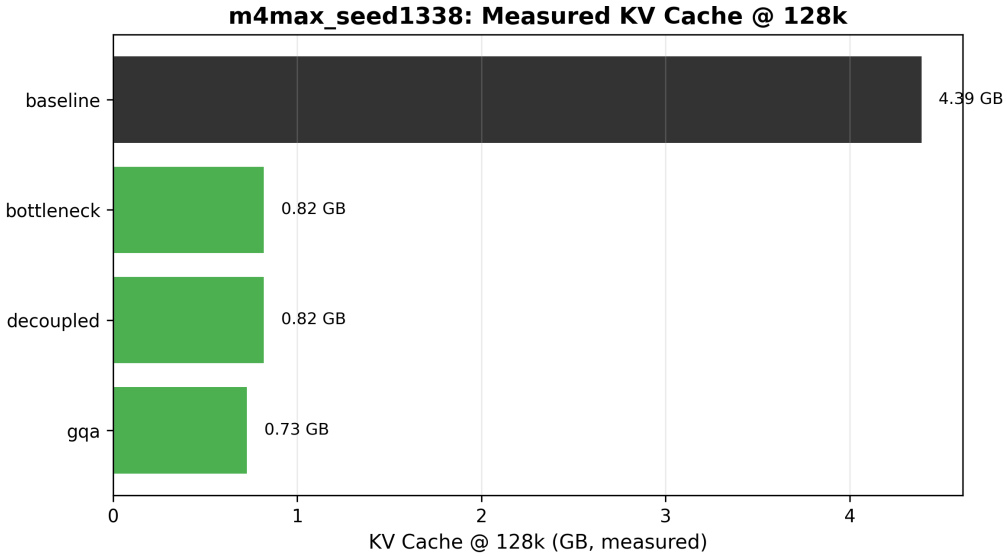


Figure 10: Measured KV-cache memory at 128k context (v29; seed 1338).

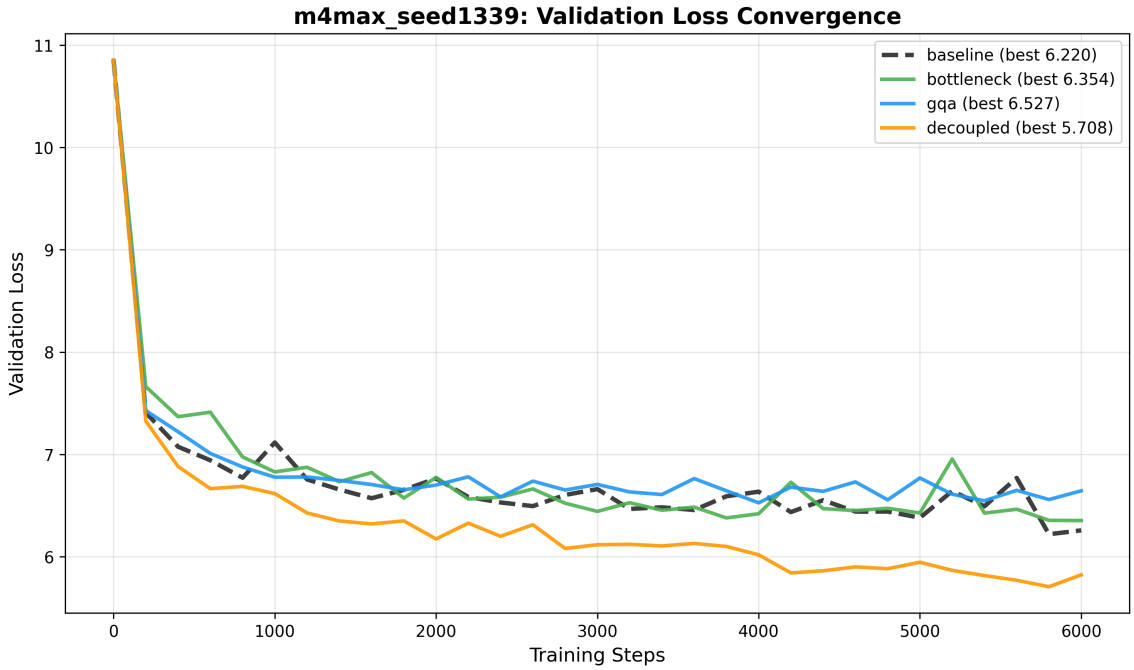


Figure 11: Validation loss curves on FineWeb-Edu (100M-token dataset; seed 1339).

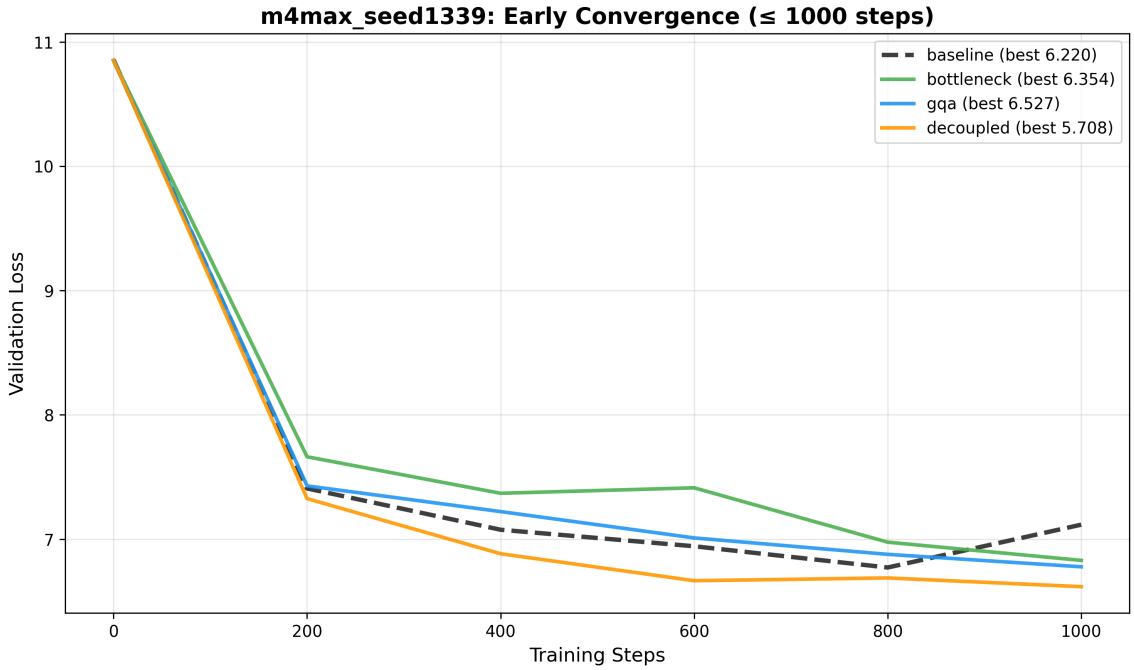


Figure 12: Early convergence view on FineWeb-Edu (seed 1339), highlighting the first phase of optimization.

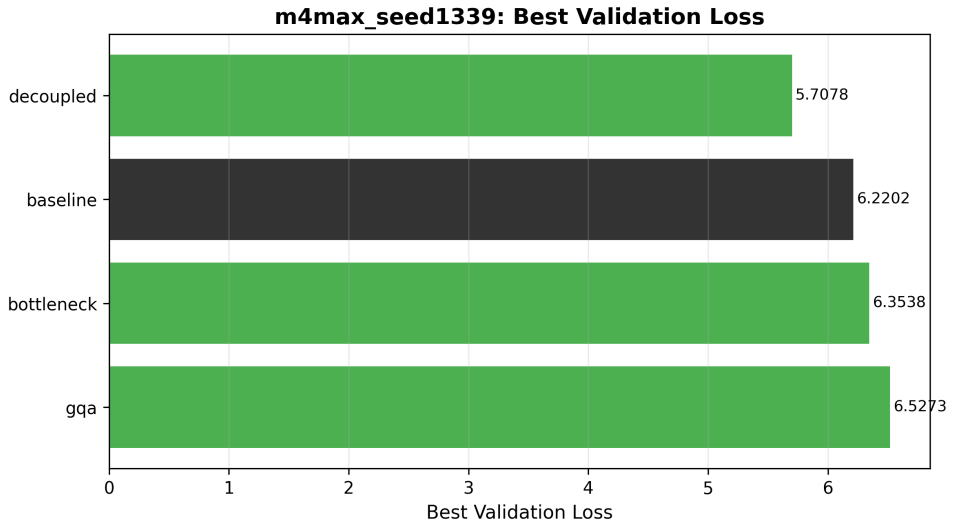


Figure 13: Final validation loss comparison on FineWeb-Edu (v29; seed 1339). Lower is better.

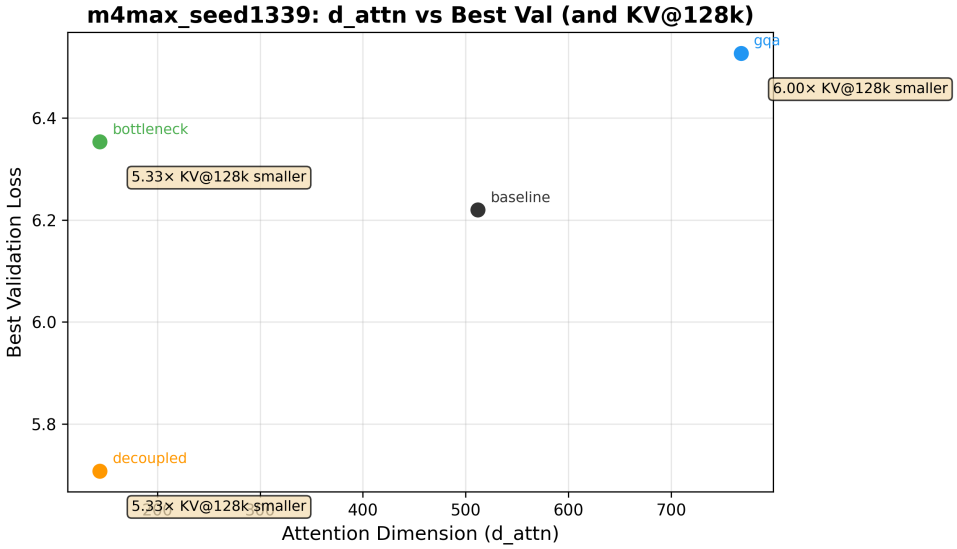


Figure 14: Quality–efficiency comparison on FineWeb-Edu (v29; seed 1339).

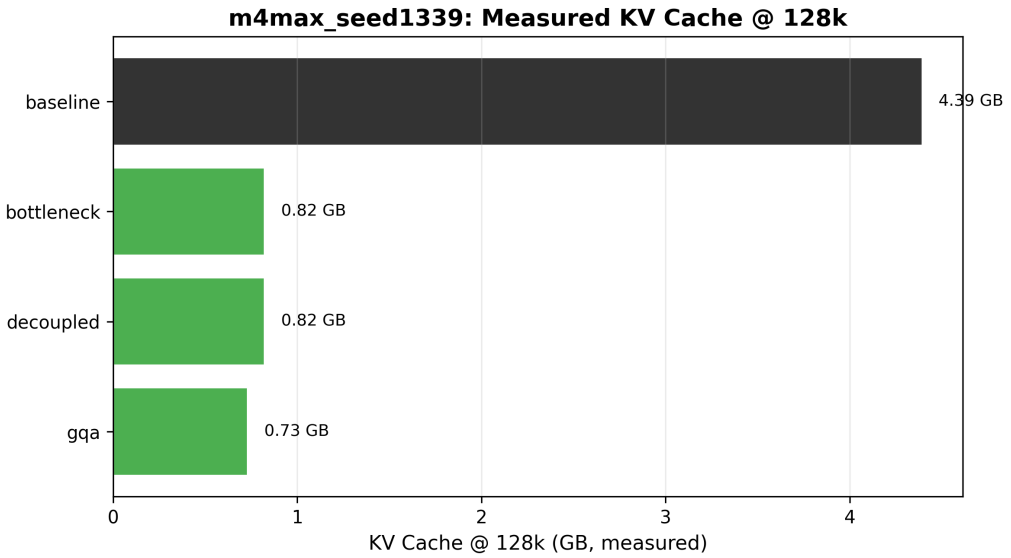


Figure 15: Measured KV-cache memory at 128k context (v29; seed 1339).

use the standard entropy effective rank:

$$\text{eRank} = \exp(H(p)), \quad p_i = \frac{\sigma_i}{\sum_j \sigma_j},$$

where $\{\sigma_i\}$ are singular values and $H(\cdot)$ is Shannon entropy. The script to reproduce this figure is included in the repository; the figure below is generated automatically when that script is run.

C Long-Context Sanity Checks

To complement the theoretical KV-cache scaling analysis, we include two lightweight long-context probes: (1) teacher-forced loss at increasing context lengths (RoPE extrapolation), and (2) a passkey "needle-in-a-haystack" next-token retrieval probe. These are not full downstream evaluations, but they help catch obvious long-context regressions and RoPE failures.

D Decoupled Ablations (v29; Seed 1337)

To support claims about optional stabilizers (null token, tied Q–K projections) and positional encoding, we report a small ablation suite for Decoupled 48/96 on FineWeb-Edu (6000 steps; seed 1337). Lower is better.

Table 4: Decoupled 48/96 ablations (FineWeb-Edu v29; seed 1337; 6000 steps).

Variant	**Best Val Loss**	**Δ vs base**
Decoupled 48/96 (base)	5.879	+0.000
− null token (`no_null`)	5.843	−0.035
+ tied Q–K (`tie_qk`)	5.926	+0.047
+ null token (`null`)	6.395	+0.516
− RoPE (`no_rope`)	6.446	+0.567

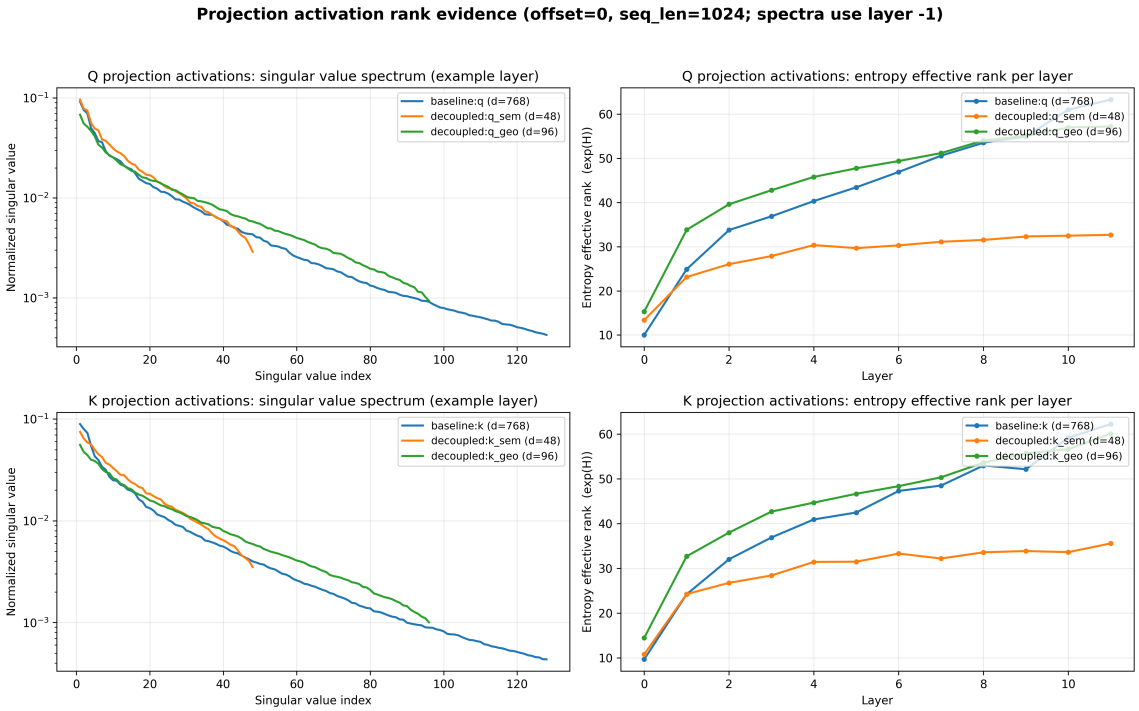
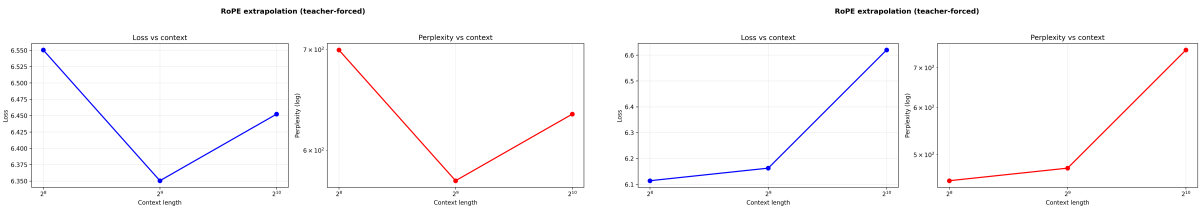
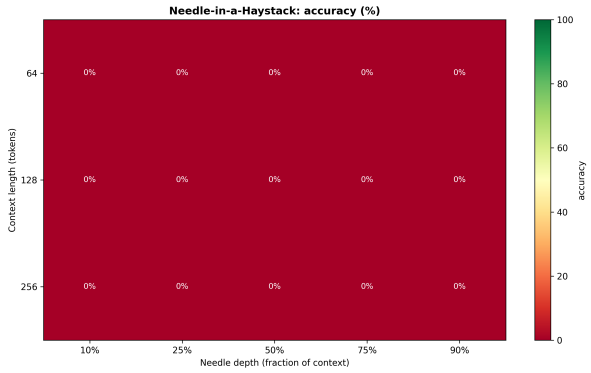


Figure 16: Singular value spectra and entropy-based effective rank for Q/K projection activations across layers (example: v29 checkpoints).

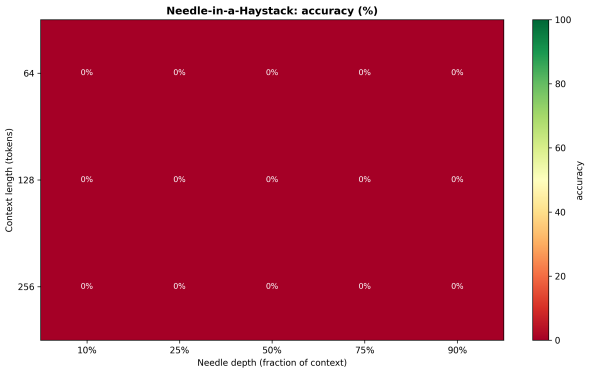


(a) Baseline (b) Decoupled

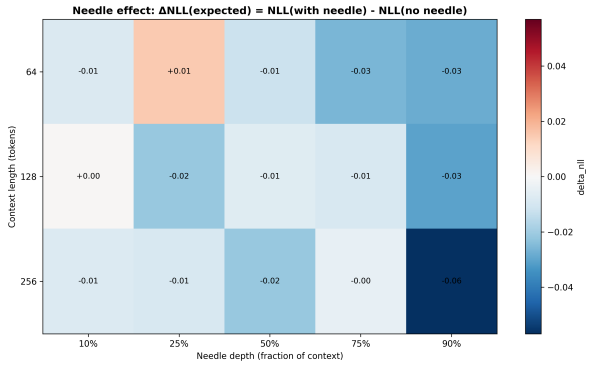
Figure 17: RoPE extrapolation probe (teacher-forced). Figures are generated by `test_rope_extrapolation_v29.py`.



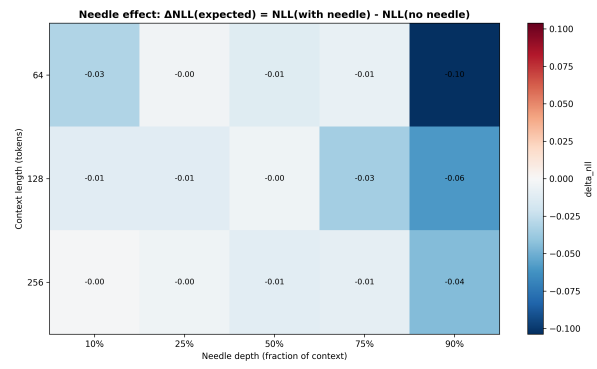
(a) Baseline



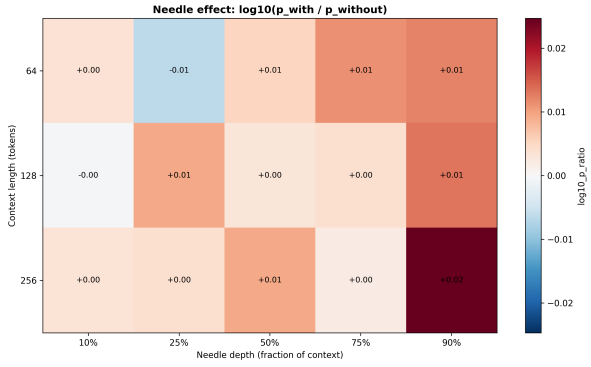
(b) Decoupled



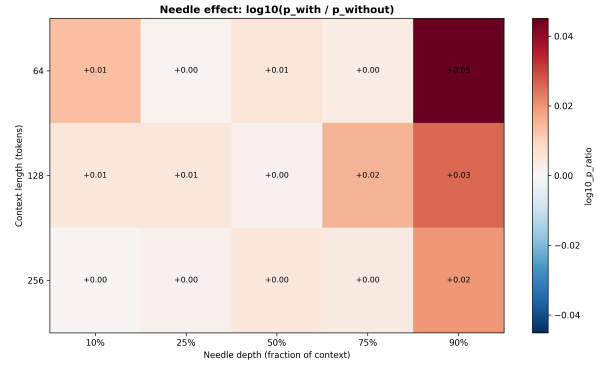
(c) Baseline: ΔNLL (with needle − no needle)



(d) Decoupled: ΔNLL (with needle − no needle)



(e) Baseline: $\log_{10}(p_{\text{with}}/p_{\text{without}})$



(f) Decoupled: $\log_{10}(p_{\text{with}}/p_{\text{without}})$

Figure 18: Passkey needle-in-a-haystack probe. Top row: strict next-token accuracy (often all-zero for base LMs). Middle/bottom: likelihood-based needle effect metrics which remain informative even when strict accuracy is zero. Figures are generated by `test_needle_haystack_v29.py`.

E Baseline Fairness Checks (v29; Seed 1337)

To address concerns that the full-rank baseline might be under-tuned under a short training horizon, we report (i) a small learning-rate sweep for the baseline at 6000 steps and (ii) a longer-horizon comparison at 12000 steps (seed 1337). Lower is better.

Table 5: Baseline LR sweep and longer-horizon comparison (FineWeb-Edu v29; seed 1337).

Run	**Steps**	**Best Val Loss**
Baseline (3×10^{-4})	6000	6.296
Baseline (2×10^{-4})	6000	6.052
Baseline (4×10^{-4})	6000	6.791
Baseline (3×10^{-4})	12000	6.133
Decoupled 48/96 (3×10^{-4})	12000	5.657

References

[1] Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *ACL*, 2021.

[2] 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. *EMNLP*, 2023.

[3] Noah Amsel, Gilad Yehudai, and Joan Bruna. Quality over quantity in attention layers: When adding more heads hurts. In *ICLR*, 2025. OpenReview: `https://openreview.net/forum?id=y9Xp9NozPR`.

[4] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[5] Srinadh Bhojanapalli et al. Low-rank bottleneck in multi-head attention models. *ICML*, 2020.

[6] David Chiang and Dani Yogatama. The rotary position embedding may cause dimension inefficiency in attention heads for long-distance retrieval. *arXiv preprint*, 2025.

[7] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Łukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers. In *ICLR*, 2021. arXiv:2009.14794.

[8] DeepSeek-AI. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024.

[9] Georgi Gerganov et al. 4-bit kv cache implementation. `https://github.com/ggml-org/llama.cpp/pull/7412`, 2024. llama.cpp PR#7412: Production Q4_0/Q8_0 KV cache support.

[10] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. *ICLR*, 2021.

[11] Coleman Hooper et al. Kvquant: Towards 10 million context length llm inference with kv cache quantization. *arXiv preprint arXiv:2401.18079*, 2024.

[12] Edward J Hu et al. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[13] Yunpeng Huang, Jingwei Xu, Junyu Lai, Zixu Jiang, Taolue Chen, Zenan Li, Yuan Yao, Xiaoxing Ma, Lijuan Yang, Hao Chen, Shupeng Li, and Penghao Zhao. Advancing transformer architecture in long-context large language models: A comprehensive survey. *arXiv preprint arXiv:2311.12351*, 2023.

[14] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *ICLR*, 2020. arXiv:2001.04451.

[15] Seijin Kobayashi, Johannes von Oswald, and João Sacramento. Weight decay induces low-rank attention layers. *NeurIPS*, 2024.

[16] Yilun Kuang, Noah Amsel, Sanae Lotfi, Shikai Qiu, Andre Potapczynski, and Andrew Gordon Wilson. Customizing the inductive biases of softmax attention using structured matrices. In *ICML*, 2025. OpenReview: `https://openreview.net/forum?id=Roc5O1ECEt`.

[17] Junyan Li, Yang Zhang, Muhammad Yusuf Hasan, Talha Chafekar, Tianle Cai, Zhile Ren, Pengsheng Guo, Foroozan Karimzadeh, Colorado Reed, Chong Wang, and Chuang Gan. Commvq: Commutative vector quantization for kv cache compression. *arXiv preprint arXiv:2506.18879*, 2025. ICML 2025 poster.

[18] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016. Introduces WikiText-2 and WikiText-103 benchmarks.

[19] Jongchan Park et al. Bam: Bottleneck attention module. *BMVC*, 2018.

[20] Yehonathan Refael, Jonathan Svirsky, Boris Shustin, Wasim Huleihel, and Ofir Lindenbaum. Adarankgrad: Adaptive gradient-rank and moments for memory-efficient llms training and fine-tuning. *arXiv preprint arXiv:2410.17881*, 2024.

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

[22] Jianlin Su et al. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

[23] Turboderp. Quantized kv cache evaluation. `https://github.com/turboderp/exllamav2/blob/master/doc/qcache_eval.md`, 2024. ExLlamaV2 implementation showing Q4 cache matches FP16 quality.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017.

[25] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

[26] Yizhou Wang et al. Attention layers add into low-dimensional residual subspaces. *arXiv preprint*, 2025. Shows attention outputs are approximately 60% low-dimensional.

[27] Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. *arXiv preprint arXiv:2007.14062*, 2020.